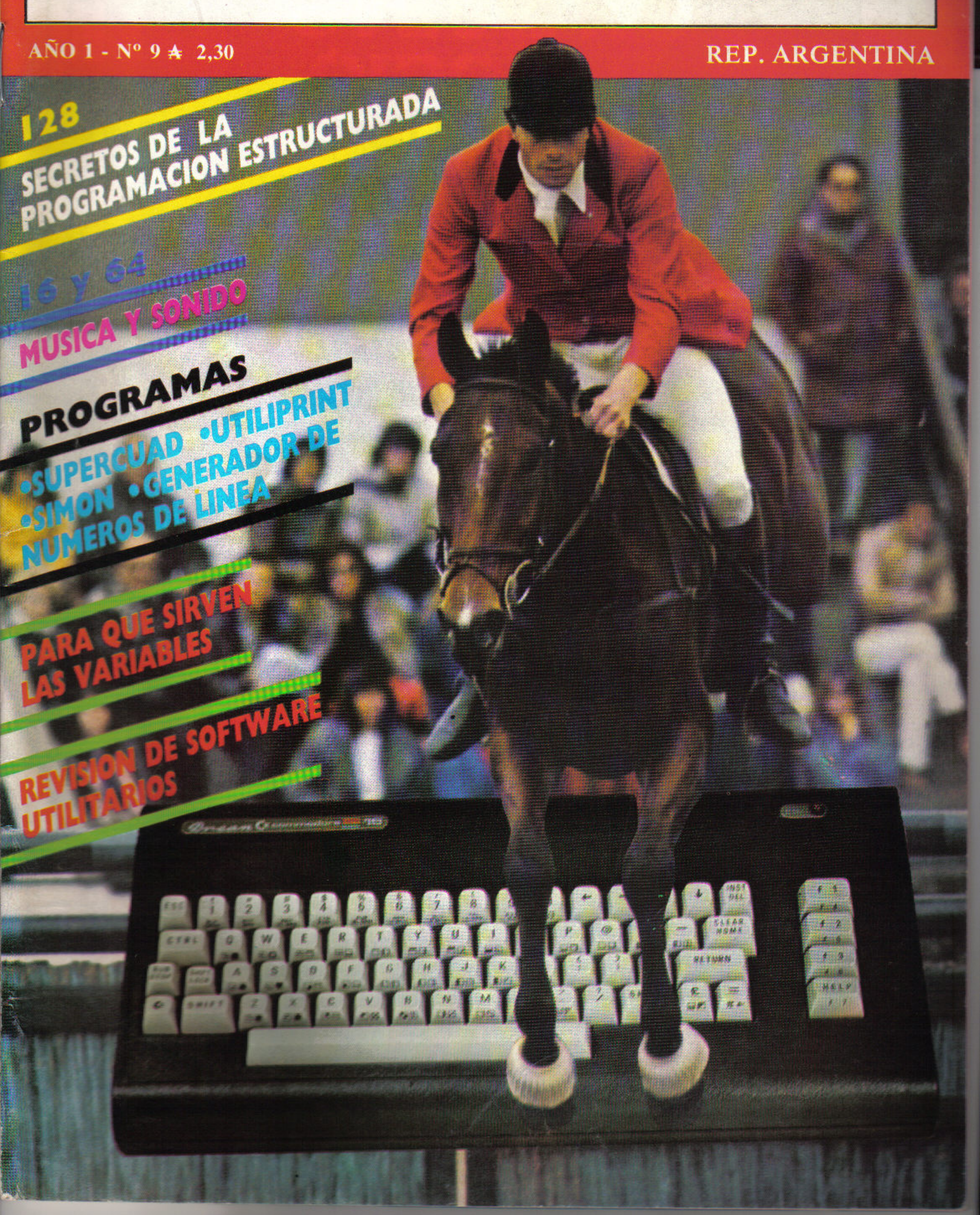
REVISTA PARA USUARIOS DE

Drean Commodore

AÑO 1 - Nº 9 ₳ 2,30

REP. ARGENTINA



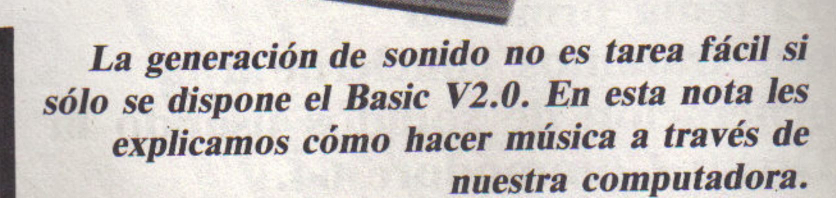
SUMARIO

NOTAS TÉCNICAS

PROGRAMAS

REVISIÓN DE SOFT

SECCIONES FIJAS



La generación de sonido no es tarea fácil si sólo se dispone el Basic V2.0. En esta nota les explicamos cómo hacer música a través de nuestra computadora.

Gracias al nuevo Basic V7.0 de la Commodore 128 tenemos herramientas sumamente potentes orientadas a la programación estructurada.





Los utilitarios vienen marchando. Uno es capaz de convertir la computadora en una pequeña redacción, el otro les permitirá procesar texto y diseñar gráficos. Pero sin olvidarnos de los juegos, los llevamos al mundo del sabotaje

Drean Commodore

AÑO 1 Nº 9 AGOSTO DE 1986

Director General
Ernesto del Castillo

Director Editorial
Cristian Pusso

Director Periodístico
Fernando Flores

Director Financiero:
Javier Campos Malbrán

Arte y Diagramación
Fernando Amengual
Tamara Migelson

Coordinador
Ariel Testori

Redacción
Cristián Parodi

Fotografia
Victor Grubicy

Departamento de Avisos
Oscar Devoto

Departamento de Publicidad
Guillermo González Aldalur

Drean Commodore es una Revista mensual editada por editorial PROEDI S.A., Paraná 720. 5º Pis. (1017) Buenos Aires. Tel.: 46-2886 y 49-7130. Reg. Nac. de la Prop. Intelectual E.T., M. Registrada.

Precio de este ejemplar. ₳ 2.30
Impresión: Calcotam. Fotocromo tapa: Columbia. Fotocomposición: Van Waveren.
Los ejemplares atrasados se venderán al precio del último número en circulación.

Distribuidor en Capital: Martino, Juan de Garay 358, P.B. Capital. Distribuidor interior: DGP, Hipólito Yrigoyen 1450, Capital Federal, T.E. 38-9266/9800.

MUSICA PARA TODOS

Seguimos explicando los conceptos básicos. El tema principal que tratamos concierne a la generación de sonidos usando la Drean Commodore 64.



No todas las computadoras hogareñas disponen de un circuito integrado tan sofisticado como lo es el SID (Sound Interface Device).

Este se encuentra dentro de la Drean Commodore 64 y permite, básicamente, poder hacer funcionar una orquesta polifónica. Como ejemplo citamos el famoso programa Kawasaky, utilitario que convierte la Drean Commodore en un sintetizador musical.

El SID dispone de tres voces, cada una de las cuales puede generar sonido en forma totalmente independiente.
Es decir que cuando una está generando un sonido dado, las otras dos pueden estar realizando otros.
El único "pero" que podemos comentar es que el Basic V2.0 (el que viene con la computadora) no contiene instrucciones orientadas al manejo del SID.
Al igual que todos los circuitos integrados, éste tiene registros que permiten su funcionamiento.
Así nosotros debemos poner en cada uno de esos registros valores determinados para que se genere un sonido.
Cada uno de estos se ejecuta teniendo en cuenta el ADSR, es decir Attack, Decay, Sustain y Release.
El Attack es el tiempo que tarda la nota en alcanzar el volumen máximo. El Decay es el tiempo que tarda en alcanzar el volumen medio.
El Sustain es el tiempo que se mantiene esa nota en el volumen medio. Finalmente el Release es el tiempo que tarda la nota en alcanzar el volumen mínimo (cero).
Cambiando el tema, pasamos a cómo programar mejor. En la nota correspondiente se explica la conveniencia de definir las variables a utilizar al comienzo del soft.
Por razones prácticas definimos a una variable como una caja que puede ir tomando valores distintos a lo largo del programa.
También cada una de ellas debe mantener una cierta coherencia con respecto a los valores almacenados.
Es decir que si una caja se usa para almacenar números (enteros o de punto flotante) no podemos almacenar caracteres.
(Recordemos que los números de punto flotante son aquellos que tienen punto decimal).
En cambio, sí podemos almacenar números en variables definidas para almacenar caracteres, ya que un número también es un caracter.
En la nota de Kernal continuamos explicando las subrutinas que utiliza la Drean Commodore 64.
Entre otras, comentamos la subrutina que se encarga de poner el cursor en una determinada posición de la pantalla.
Recuerden que una subrutina es un procedimiento que se ejecuta más de una vez. Por este motivo sólo lo escribimos una vez y lo ejecutamos cuando lo necesitamos.

Nueva Commodore 64

La empresa Commodore ha lanzado en los Estados Unidos la nueva Commodore 64 C. Tiene el diseño de la Commodore 128. Internamente es igual a la C-64. La única diferencia es que esta puede trabajar opcionalmente con el sistema operativo GEOS.



Aplicaciones

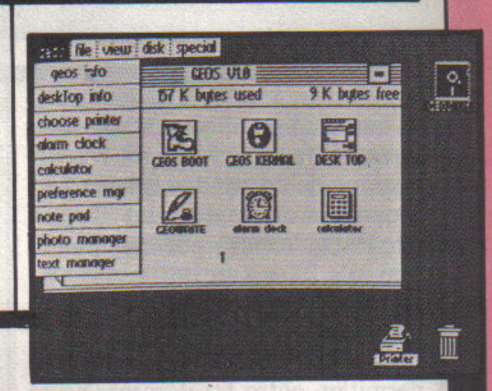
La firma TEX (americana) creo un programa que gobierna un robot (también diseñado por dicha firma) usando para ello el port del usuario y un radio control.
Este robot en etapa experimental (todavía no se ha comercializado) es capaz de tomar pequeños objetos, trasladarlos, etc.
Lo interesante del proyecto es que el usuario puede especificar una serie de módulos, los cuales constituyen los movimientos del robot.
A través de instrucciones básicas, es decir aquellas que ya están definidas, se crean estos módulos.

Juego

GREEN BERET (Boinas Verdes) es uno de los últimos juegos que acaban de salir para la Drean Commodore 64.
El estilo es del tipo "Comando", aunque aseguran que muy superior. La cantidad de pantallas y sonidos que posee son sumamente sofistificados.
Más datos, llamar al 35-6106.
Esperamos que en próximos números podamos realizar la revisión de este estupendo juego.

Manual para C-128

"Guía de Referencia del Programador" explica en forma detallada cómo es la C-128 por dentro, describe las subrutinas del sistema operativo para que el usuario pueda acceder fácilmente a ellas, ofrece detalles referentes a los circuitos de la computadora, explica a fondo cómo trabajar con sonido y música, etc., y no descuida el modo CP/M. Mayor información, llamar al 97-2879.



Sistema operativo

La empresa Berkley Software creó un nuevo sistema operativo para la Commodore 64.
Su nombre es GEOS, siglas de Graphic Enviroment Operating Sistem.
Ya está en la Argentina y es ciento por ciento compatible con la Drean Commodore 64.
El sistema incluye entre otras cosas, un procesador de texto (llamado GEOWRITE) y un utilitario para desarrollar gráficos en alta resolución (GEOPAINT).
En Revisión de Software de este número hacemos un análisis a fondo de este producto.

Lanzamientos

Drean, licenciataria en Argentina de Commodore, expandirá este año sus actividades, "atacando" el mercado de los Professional Computers (PC), además del de Home.

Entre los proyectos, figuran los lanzamientos de la 128, la Amiga y los PC-10 y PC-20, estos últimos compatibles con la línea IBM.

MUSICA Y SONIDO

La generación de sonido no es tarea fácil si sólo se dispone el Basic V2.0. En esta nota les explicamos cómo hacer música a través de nuestra computadora.



Como todos saben el Basic de la Drean Commodore 64 no dispone de comandos orientados al manejo del sonido.
Programas como, por ejemplo, Simon's Basic (un Basic mejorado para esta computadora) sí le ofrecen al programador instrucciones que le permiten, entre otras cosas, hacer música. El Basic V2.0 sólo nos ofrece, básicamente, una instrucción para la generación de sonido: la sentencia POKE.
Esto significa que para realizar música necesitamos poner ciertos valores en determinadas direcciones de memoria.
La Drean Commodore 64 dispone de un chip que se encarga de realizar los distintos efectos sonoros. Su nombre: SID son las siglas de Sound Interface Device.
Internamente está formado por 29 registros enumerados de 0 a 28.
Cada registro realiza una determinada función.
Combinándolos se obtiene la generación de un determinado sonido.
Pero antes de describir las funciones exactas de ellos veamos, primero, los rasgos generales del chip SID.
La característica más importante es que dispone de tres voces totalmente independientes. Es decir que, por ejemplo, mientras una ejecuta una determinada nota las otras pueden realizar el acompañamiento.
Internamente contiene filtros que permiten seleccionar una porción de la nota generada.
Cada una de estas voces (denominadas V1, V2 y V3) deben tener una serie de parámetros para que se pueda ejecutar un sonido.
Estos se refieren al valor de la frecuencia a ejecutar, el tipo de onda, el ancho del pulso y finalmente el **Attack, Decay, Sustain** y **Release (ADSR).**
En la figura 1 se muestra el diagrama que corresponde a la ejecución de una nota.
Cuando ésta se reproduce se inicia el **Attack**, es decir el tiempo en que tarda en alcanzar el volumen máximo.
Luego de llegar a este "pico" comienza el **Decay**, es decir el tiempo que tarda en alcanzar el volumen medio.
A continuación se inicia el **Sustain** en donde se sostiene ese nivel de sonido durante un cierto tiempo.
Finalmente se ejecuta el período **Release**, el cual representa el tiempo en que se tarda en alcanzar el volumen mínimo (cero).
Cada nota o frecuencia que se ejecuta siempre pasa por el proceso **ADS**. La nota culmina su ejecución cuando se inicia el ciclo **Release.**

Tabla 1

Valor	Tiempo de Attack	Tiempo de D/R
0	2 ms	6 ms
1	8 ms	24 ms
2	16 ms	48 ms
3	24 ms	72 ms
4	38 ms	114 ms
5	56 ms	168 ms
6	68 ms	204 ms
7	80 ms	240 ms
8	100 ms	300 ms
9	250 ms	750 ms
10	500 ms	1.5 s
11	800	2.4 s
12	1 s	3 s
13	3 s	9 s
14	5 s	15 s
15	8 s	24 s

Esta tabla se debe comprender de la siguiente manera: si se pone un valor de 0 para Attack, la nota tardará 2 milisegundos en alcanzar el pico máximo.
O si se selecciona un valor de 12 para Decay, la nota tardará 3 segundos en alcanzar el nivel medio.

Tabla 2

SID=54272	b7	b6	b5	b4	b3	b2	b1	b0	VOZ 1
SID	F7	F6	F5	F4	F3	F2	F1	F0	Frecuencia baja
SID+1	F15	F14	F13	F12	F11	F10	F9	F8	Frecuencia alta
SID+2	A7	A6	A5	A4	A3	A2	A1	A0	Ancho pulso bajo
SID+3	—	—	—	—	A11	A10	A9	A8	Ancho pulso alto
SID+4	ruido	⊓⊔⊓	/|/|	/\/\	test	ring mod	sync	gate	Registro de control
SID+5	AT3	AT2	AT1	AT0	DE3	DE2	DE1	DE0	Attack / Decay
SID+6	SV3	SV2	SV1	SV0	RE3	RE2	RE1	RE0	Sustain / Release
									VOZ 2
SID+7	F7	F6	F5	F4	F3	F2	F1	F0	Frecuencia baja
SID+8	F15	F14	F13	F12	F11	F10	F9	F8	Frecuencia alta
SID+9	A7	A6	A5	A4	A3	A2	A1	A0	Ancho pulso bajo
SID+10	—	—	—	—	A11	A10	A9	A8	Ancho pulso alto
SID+11	ruido	⊓⊔⊓	/|/|	/\/\	test	ring mod	sync	gate	Registro de control
SID+12	AT3	AT2	AT1	AT0	DE3	DE2	DE1	DE0	Attack / Decay
SID+13	SV3	SV2	SV1	SV0	RE3	RE2	RE1	RE0	Sustain / Release
									VOZ 3
SID+14	F7	F6	F5	F4	F3	F2	F1	F0	Frecuencia baja
SID+15	F15	F14	F13	F12	F11	F10	F9	F8	Frecuencia alta
SID+16	A7	A6	A5	A4	A3	A2	A1	A0	Ancho pulso bajo
SID+17	—	—	—	—	A11	A10	A9	A8	Ancho pulso alto
SID+18	ruido	⊓⊔⊓	/|/|	/\/\	test	ring mod	sync	gate	Registro de control
SID+19	AT3	AT2	AT1	AT0	DE3	DE2	DE1	DE0	Attack / Decay
SID+20	SV3	SV2	SV1	SV0	RE3	RE2	RE1	RE0	Sustain / Release
									FILTROS
SID+21	—	—	—	—	—	FC2	FC1	FC0	Frecuencia de corte baja
SID+22	FC10	FC9	FC8	FC7	FC6	FC5	FC4	FC3	Frecuencia de corte alta
SID+23	RES3	RES2	RES1	RES0	Filtex	FILT3	FILT2	FILT1	Resonancia
SID+24	3OFF	HP	BP	LP	Vol3	Vol2	Vol1	Vol0	Volumen
									MISCELANEOS
SID+25	PX7	PX6	PX5	PX4	PX3	PX2	PX1	PX0	Pot X
SID+26	PY7	PY6	PY5	PY4	PY3	PY2	PY1	PY0	Pot Y
SID+27	07	06	05	04	03	02	01	0o	Oscilador/Random
SID+28	37	E6	E5	E4	E3	E2	E1	E0	Env3

Con lo que respecta al tipo de onda, la figura 2 representa los distintos tipos que el SID puede generar.
Estos son **onda triangular, diente de sierra** y **onda cuadrada.** Para esta última es posible indicar el ancho del pulso y también generar ruido blanco. Cada una de estas ondas determinan la característica del sonido. Nosotros, a través de un programa, iremos seleccionando los distintos tipos de ondas para determinar la calidad del sonido.

Los primeros pasos

Vayamos pues a los ejemplos concretos para que la teoría antes descripta pueda ver resultados positivos.
El proceso que a continuación explicaremos es idéntico para cada una de las tres voces. Lo único que cambian son los números de registros.
Nosotros utilizaremos la **V1.**
Lo primero que debemos realizar es determinar el **ADSR.** Para eso se usan los registros 5 y 6, correspondientes a la **V1.**
Los 4 bits más altos (7, 6, 5 y 4) del registro 5 se encargan de determinar el **Attack** mientras que los cuatro bits más bajos (3, 2, 1 y 0) determinan el **Decay.**
En el registro 6 sucede lo mismo con la salvedad que los cuatro bits más altos determinan el **Sustain** mientras que los más bajos determinan el **Release.**
Para cada uno de los procesos (**A, D, S** o **R**) se permiten números entre 0 y 15. Así, un **Attak** de 0 ocasionará que la nota alcance su nivel máximo en un tiempo muy pequeño mientras que con 15 tardará mucho más.
La tabla 1 representa los tiempos exactos en función de los valores ingresados en estos registros.
Luego de haber seleccionado el **ADSR,** debemos determinar la frecuencia o nota que se ejecutará.
Para ello se usa el registro número 0 y 1. Como la frecuencia puede tomar valores de 0 a 65535 es necesario usar dos bytes para poder representarla (recuerden que en

un solo byte sólo se pueden poner valores comprendidos entre 0 y 255).

En caso de usar notas distintas a las que figuran en el apéndice P del manual del usuario de la Drean Commodore 64, se debe realizar la siguiente ecuación:

$F_o = F_i / 0.06097$.

Luego se debe pasar este número a formato alto-bajo utilizando la siguiente ecuación:

$FA = INT (F_o/256)$
$FB = F_o - 256*FA$

Hecho esto debemos **POKE**arlos en memoria.

Luego de establecer la frecuencia, debemos indicar el volumen. Esto se realiza usando los cuatro bits más bajos del registro 24. De esta manera el rango de volumen oscila entre 0 (mínimo) y 15 (máximo).

Finalmente seleccionamos el tipo de onda e iniciamos el "getting". **Getting** significa iniciar el proceso **Attack, Decay y Sustain.**

Para eso se usa el registro número 4. Cada uno de los bits que lo constituyen realizan una determinada función.

El bit 0 representa el **GATE**. Con un "1" se inicia el ciclo **ADS** mientras que con un "0" se inicia el ciclo **R**.

El bit 1 representa el **SYNC**. Su función es la de sincronizar la voz 1 con la 3. De esta manera se realizan buenos efectos. La única salvedad es que la **V3** debe tener una frecuencia distinta de cero.

Es decir que no hace falta poner algún otro parámetro en **V3** para lograr este efecto.

Para activar el **SYNC** se debe poner el bit 1 a "1". Para desactivarlo basta con ponerlo a "0".

El bit 2 es el **RING MOD**. Con un "1" se activa mientras que con un "0" se desactiva.

A través de ésta también se generan efectos especiales. Para utilizarla es necesario seleccionar la onda triangular del **V1** y la frecuencia de la **V3** debe ser distinta de cero.

El bit 3 se llama **TEST**. Se utiliza para sincronizar las tres voces y eventos externos.

Figura 1

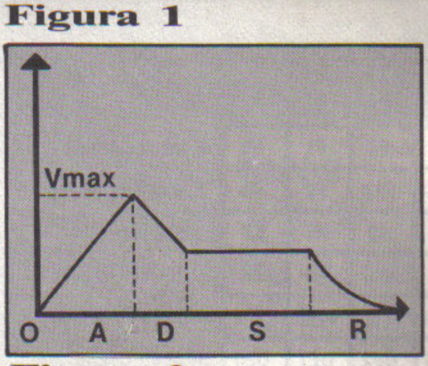
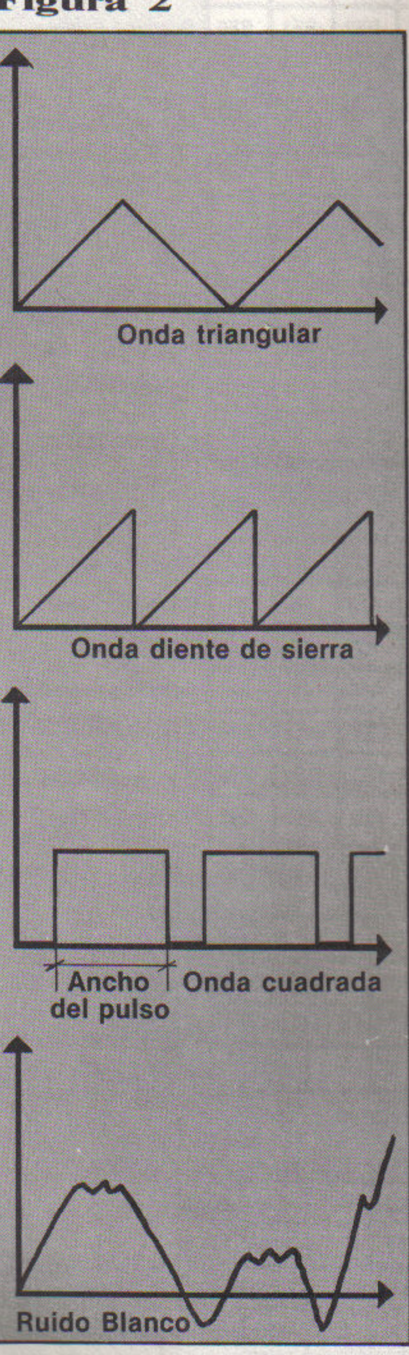




Figura 2



El bit 4 representa el tipo de onda **triangular**. Cuando está a "1" se selecciona mientras que con un "0" se desactiva.

El bit 5 selecciona el tipo de onda **diente de sierra.**

El bit 6 selecciona el tipo de onda **cuadrada**. En caso de seleccionarse, se debe indicar en los registros 2 y 3 el ancho del pulso.

En el registro 2 se coloca la parte baja mientras que en los primeros 4 bits (0, 1, 2 y 3) se pone la parte alta. Los cuatro bits más altos no se utilizan.

El bit 7 representa el generador de **ruido blanco.**

La tabla 2 representa el esquema de todos los registros del SID. Ustedes verán que también existe la posibilidad de filtrar las frecuencias generadas por el sintetizador. Esto se realiza usando los distintos filtros del SID.

De todas maneras volveremos a tomar este tema en particular en futuros números.

Los pasos finales

Finalmente les dejamos la siguiente tabla para que puedan desarrollar música en su Drean Commodore 64

1) Limpiar todos los registros del **SID**
2) Determinar el **ADSR**
3) Determinar volumen
4) Determinar frecuencia (no olvidar la fórmula si se trata de frecuencias "extrañas")
5) Determinar tipo de onda y prender **GATE**. Si la onda seleccionada es cuadrada determinar ancho de pulso.
6) Poner un retardo que establezca la duración de la ejecución de la nota.
7) Pone **GATE** a 0 (apaga sonido)

Los programas correspondientes al listado 1 y al listado 2 son ejemplo de generación de sonidos.

El primero de ellos realiza una ejecución normal de una frecuencia dada.

El segundo simula una computadora de "dibujitos animados".

DREAN COMMODORE 16 Y 64

Listado 1

```
10 REM GENERACION DE TONO
30 SID=54272:REM DIRECCION INICIO SID
40 POKESID+24,15:REM VOLUMEN
50 POKESID,9:REM FRECUENCIA BAJA
60 POKESID+1,12:REM FRECUENCIA ALTA
70 POKESID+5,64:REM ATTACK/DECAY
80 POKESID+6,64:REM SUSTAIN/RELEASE
90 POKESID+4,17:REM SIERRA Y GATE ON
100 FORFE=1TO4000STEP10:F=FE/.06097
110 FA=INT(F/256):FB=F-256*FA:REM CALCULA PARTE ALTA Y BAJA
120 POKESID,FB:POKESID+1,FA:REM PONE NUEVA FRECUENCIA
130 NEXT
140 POKESID+4,0:REM GATE OFF-APAGA SONIDO
150 END
```

Listado 2

```
10 REM EJEMPLO DE GENERACION DE SONIDO
20 REM LABORATORIO ELECTRONICO
30 SID=54272:REM DIRECCION INICIO SID
40 POKESID+24,15:REM VOLUMEN
50 POKESID,3:REM FRECUENCIA BAJA
60 POKESID+1,12:REM FRECUENCIA ALTA
70 POKESID+5,64:REM ATTACK/DECAY
80 POKESID+6,64:REM SUSTAIN/RELEASE
90 POKESID+4,21:REM SIERRA/RING/GATE
100 FORE=1TO4500
110 POKESID,INT(RND(1)*255):POKESID+1,INT(RND(1)*100)
120 POKESID+14,INT(RND(1)*255):POKESID+15,INT(RND(0)*255)
130 POKESID+1,INT(RND(1)*100)
140 NEXT
150 POKESID+4,0:REM GATE OFF
160 END
```

EN LA DREAN COMMODORE 16

Para generar sonido en la Drean Commodore 64, se debe tener pleno conocimiento de los registros del SID para poder realizar música y sonidos.
Gracias al Basic 3.5 de la Drean Commodore 16, la generación de tonos musicales es mucho más sencilla ya que existen comandos orientados a tal fin.
Aquí ya no se necesita saber cuál registro determina el volúmen o cual determina la frecuencia.
A diferencia del SID de la Drean Commodore 64, ésta computadora no puede generar sonidos con ADSR. La razón es que, en ella, hay un solo chip (el TED) que se encarga de hacer todas las funciones de la computadora como ser generar imágenes, sonidos, etc.
La Drean Commodore 16 tiene tres voces. Las dos primeras producen tonos mientras que la segunda sólo ruido blanco.
La sentencia fundamental que dispone es SOUND. Aquí se debe especificar el número de voz que se utilizará (de 1 a 3), la nota o frecuencia (valores comprendidos entre 0 y 1015) y, finalmente, la duración de la misma (entre 0 y 65535)
Un dato interesante es que optar por una duración máxima

Listado 3

```
10 REM EJEMPLO DE SONIDO MUSICAL
20 FORI=1TO200
30 V1=INT(RND(1)*1000)
40 V2=INT(RND(1)*1000)
50 SOUND1,V1,20
60 SOUND2,V2,40
70 NEXTI
80 END
```

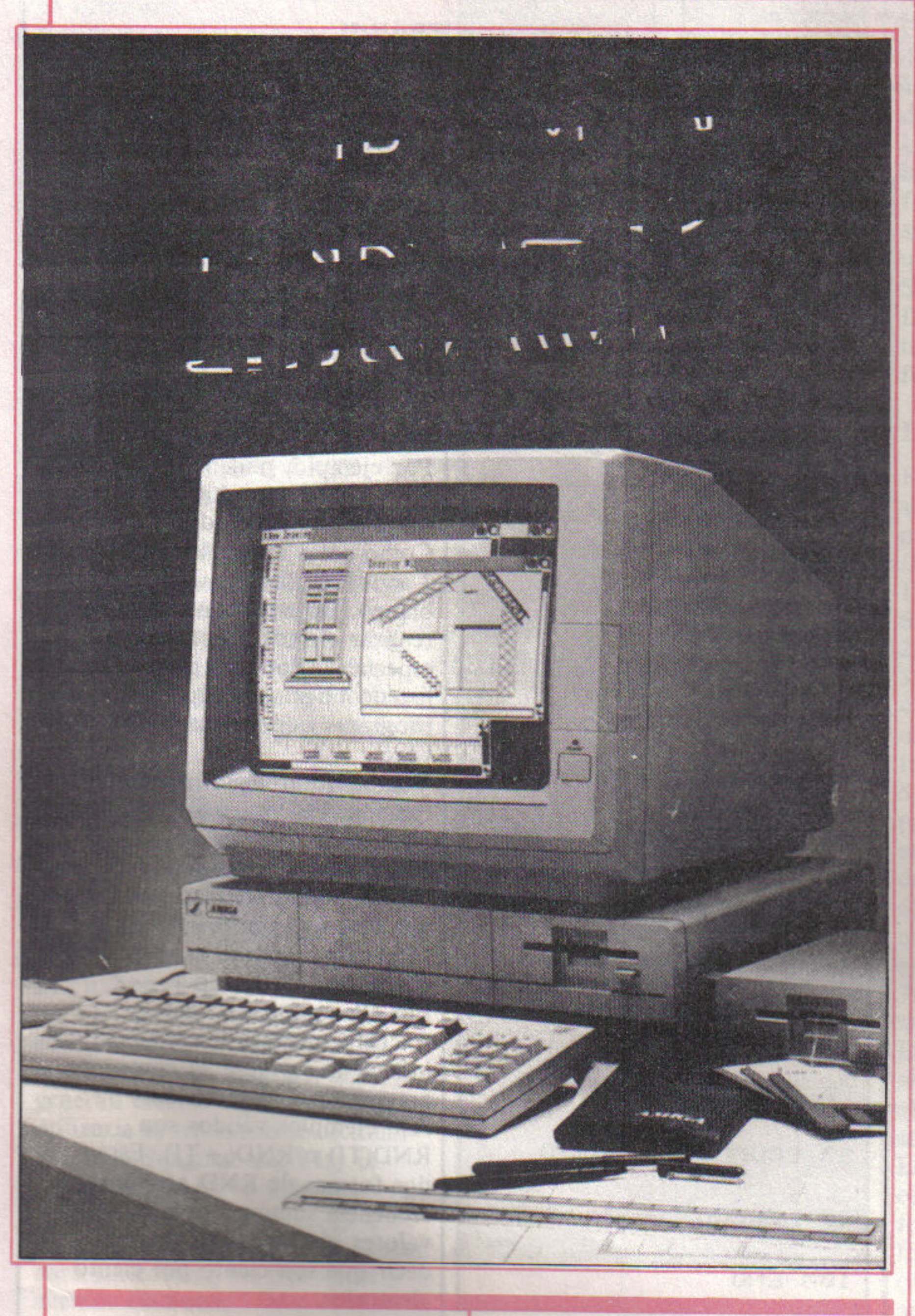
Listado 4

```
10 REM RUIDO
20 FORCO=1TO400
30 V1=INT(RND(1)*1000)
40 V2=INT(RND(1)*1000)
50 DU=INT(RND(1)*40)
60 SOUND1,V1,DU
70 SOUND2,V2,DU+10
80 SOUND3,V3,DU+20
90 NEXTCO
100 END
```

(65535) equivale a que la nota se ejecute durante 16 minutos aproximadamente.
El otro comando que se utiliza es el VOL que establece el volumen de la nota.
Su sintáxis es VOL seguido por un número comprendido entre 0 (mínimo) y 8 (máximo).
Por ejemplo, podemos hacer un laboratorio computarizado como se indica en el listado 3
Como antes mencionamos, la voz 3 se utiliza para generar ruido blanco. De esta manera podemos realizar efectos sonoros especiales. Por ejemplo en el listado 4 hemos hecho un programa que mezcla las tres voces.
Como habrán visto, utilizamos la función RND para generar las frecuencias al azar.
Esta función es, en realidad, pseudo aleatoria. Esto significa que luego de un tiempo la secuencia de números se volverá a repetir.
Por ello, el método más frecuentemente utilizado es el de utilizar la variable TI como argumento de la función.
Así, ejemplos válidos son RND(TI) o RND(−TI). Estas dos formas de RND generan una secuencia sumamente amplia de valores, de los cuales podemos decir que son ciento por ciento aleatorios.

AMIGA

Ya revolucionó el mercado norteamericano. Ahora llegó a la Argentina. Esta nueva computadora personal tiene características sobresalientes. La más importante es que es compatible con la PC de IBM.



La Commodore Amiga apareció en el mercado norteamericano el otoño del año pasado, a través de una espectacular presentación que se llevó a cabo en uno de los más importantes salones de exposición de los Estados Unidos.
Desde ese momento la popularidad de la Amiga aumentó en forma exponencial. Poco a poco comenzaron a aparecer clubes de usuarios de esta nueva PC.
Cuenta la leyenda, que el nombre de "Amiga" se debe a que en Estados Unidos los adolescentes identificaron a la C-64 con el apodo de "Unfriendly" (algo así como no amigable) debido a la cantidad de POKEs que se deben hacer para trabajar con música, video, etc.
En compensación, Commodore Bussines diseñó esta estupenda máquina facilitando todo.
En la Argentina ya es moneda común verla en algunos negocios. Inclusive tenemos conocimiento de un club de usuarios para esta computadora.
La popularidad de la amiga también se debe a su compatibilidad con la PC de IBM, gracias a que dispone de una interface que le permite esa ventaja.
En lo que respecta a nueva tecnología, presenta muchas novedades. Empezando con el microprocesador (la unidad central de procesamiento) que es un chip Motorola 68000 de 16 bits de datos y 32 bits de direcciones.

Esto significa que los programas se ejecutan a una velocidad de procesamiento muy alta.

La memoria RAM básica es de 256 Kbyte que se pueden expandir internamente a 512 Kbyte. Externamente se puede expandir a 8 Mbyte.

A través de la inserción de un disco duro (dispositivo de almacenamiento masivo de información), la capacidad de almacenamiento de la Amiga es de unos 20 Mbyte.

En palabras más fáciles, esto quiere decir que disponemos de memoria suficiente como para almacenar miles de diccionarios enciclopédicos.
El sistema se puede describir como tres módulos independientes.
El primero es donde se encuentra la memoria principal, la unidad central de procesamiento y una unidad de minidiscos de 3 1/2" cuya capacidad es de 880 Kb.
El segundo módulo es el teclado y, finalmente, el último es el monitor color, aunque también es posible poner un televisor común.

El primer módulo, llamado la Unidad del Sistema y en donde se encuentra la memoria y el chip 68000, posibilita su expansión.
Por ejemplo, podemos insertar un modem trabajando a 2400 Baudios o el disco duro de 20 MByte.
A través del modem podemos



enlazarla con cualquiera de las grandes bases de datos.
También se dispone de un port Centronics y un port serie para conectar cualquier tipo de impresora.
Recordemos que la interface que acompaña a la mayoría de los equipos Commodore es tipo serie. Es decir que sólo podemos conectar impresoras serie.
Por otro lado existe en el mercado mundial mayor cantidad de impresoras paralelo que serie.
Gracias a estas dos nuevas interfaces podremos interconectar cualquier impresora.
La port serie también se puede utilizar para conectar la interface MIDI (Musical Instrument Digital Interface).
A través de ella podemos controlar instrumentos musicales utilizando la Commodore Amiga como órgano de control.

El teclado de la Amiga consta de 89 teclas y dispone de 10 teclas programables por el usuario.
Tiene un teclado numérico (Keypad) conteniendo 13 teclas.
El sistema operativo está constituido por tres grandes subsistemas. El primero de ellos es el Kickstart que contiene al sistema operativo básico, el Workbench con el Amiga Dos y finalmente el Intuition.
El Workbench le posibilita al usuario trabajar con el mouse (especie de joystick que mueve el cursor por toda la pantalla), seleccionando icons (gráficos denotando una función en especial), ventanas, etc.
Este módulo permite, además, copiar archivos, correr programas y trabajar fácil y rápidamente con otras funciones del Amiga Dos.
Nuestra amiga nos dejará ejecutar varios programas al mismo tiempo, lo que se conoce con el nombre de "Multitaskink". Gracias a ello podemos ordenar alfabéticamente la información almacenada en un archivo mientras realizamos operaciones de inserción y eliminación de registros en otro.
En otras computadoras, estas tareas las debemos hacer separadamente, lo cual nos llevaría el doble de tiempo.
Con respecto a las posibilidades gráficas y sonoras, esta computadora posibilita, como dijimos antes, generar 4096 colores distintos y realizar sonidos estereofónicos.

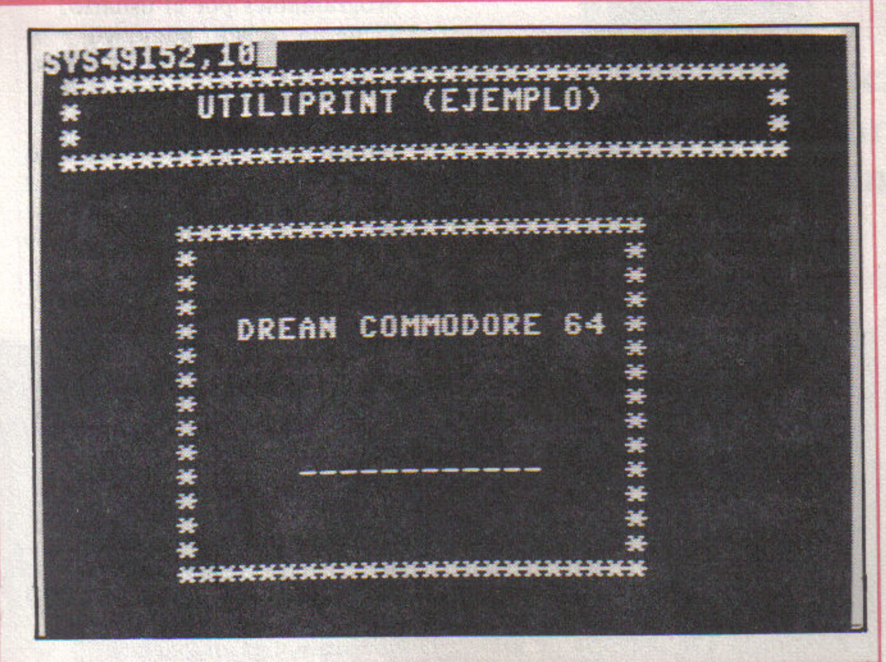
La parte gráfica puede manejar cuatro modos. El primero de ellos trabaja con una resolución de 640 x 400 pixel (recordemos que un pixel es la unidad más pequeña que se puede prender o apagar en la pantalla) junto con 16 colores. El segundo modo lo hace con 640 x 200 con 16 colores y el tercero en 320 x 200, con 32 colores.

En cualquiera de los tres modos anteriores es posible elegir un color de los 4096.

CARACTERISTICAS PRINCIPALES

Memoria: 256 kb de RAM protegida usada por el sistema operativo. La RAM es internamente expandible a 512 kb, externamente a 8 Mb.
Video: Monitor RGB color o TV comercial.
Gráficos: Cuatro resoluciones: 640 x 400, 16 colores; 640 x 200, 16 colores; 320 x 200, 32 colores; modo especial.
Sonido: Sintetizador de sonido estéreo con cuatro canales. Permite generar voz humana bajo control de software.
Teclado: 89 teclas, incluyendo Keypad con 13 teclas. Diez teclas de función, control del cursor.
Interface: Dos puertas para mouse o joystick. Slot para expansión de RAM, salidas derecho e izquierdo del sintetizador del sonido, interface serie programable, interface Centronics, Disco externo. Dentro de la unidad de procesamiento se encuentra una unidad de minidiscos de 3 1/2" con capacidad de 880 kb.

UTILIPRINT



Tipo: Utilitario
Comp.: DC-64
Conf.: Básica

Siempre nos ha pasado, en mayor o menor grado, tener que realizar comentarios explicativos durante la ejecución de un programa.
Generalmente estos comentarios se realizan usando la sentencia PRINT seguida con las líneas que formarán el texto.
Por eso, cuando los tipeamos debemos comprobar a cada momento cómo sale ese texto cuando se ejecute el programa.
De esta manera evitamos que las palabras salgan cortadas o que nuestro texto esté descentrado.
Para el desarrollo de los juegos también es necesario realizar los escenarios que formarán parte de él.
A tal fin usamos los caracteres gráficos de la Drean Commodore 64.

Aquí también puede ocurrir que no salgan como nosotros queremos.
Para evitar estos inconvenientes, les ofrecemos el UTILIPRINT.
Este programa nos permite diseñar la pantalla deseada convirtiéndolo a sus correspondientes PRINT.
Es decir que UTILIPRINT pone al final de nuestro programa todas las sentencias PRINT necesarias para formar la pantalla definida por nosotros.
Entonces los pasos a seguir son:
1) Tipeen el programa correspondiente y luego asegúrense de grabarlo en disco o en cassette ya que éste se autoborra.
2) Ejecuten el programa.
3) Borren la pantalla y comiencen a diseñar la nueva pantalla a partir de la segunda fila (línea número 2).
En ningún momento opriman RETURN. Para moverse por la pantalla sólo utilicen las teclas que mueven el cursor.
4) Luego de completar el diseño, suban con el cursor hasta la primera línea y allí, sólo allí, tipeen SYS 49152, INC. INC es el incremento de los números de línea en donde estarán los PRINT que formarán esa pantalla (acá sí opriman RETURN al final del SYS).
A continuación UTILIPRINT pondrá a partir de la última línea las sentencias PRINT junto con los caracteres ingresados anteriormente.
Es decir que, por ejemplo, si nuestro programa termina con la línea 100 y nosotros hacemos SYS 49152,10, los PRINT se pondrán en 110, 120, 130, etc.
Otra de las características de este programa es que permite utilizar caracteres en video inverso y en color.

VARIABLES UTILIZADAS

I: Puntero de carga en memoria.
A: Toma las instrucciones en decimal.
X: Sumatoria total de las instrucciones (cheksum)

DESCRIPCION DEL PROGRAMA

Línea	Descripción
100	Comienzo de lectura del programa escrito en lenguaje máquina.
110	Verificación de los valores POKEados en memoria.
120	Borra el programa en memoria.
130-620	DATAS correspondientes a los valores en decimal de las instrucciones del programa.

```
100 FORI=49152TO49649:READA:X=X+A:POKEI,A:NEXT
110 IFX<>55784THENPRINT"ERROR EN DATAS. VERIFIQUE
VALORES":STOP
120 PRINT"UTILIPRINT ACTIVADO":NEW
130 DATA  32,155,183,142, 85,  2, 32,228,193, 32
140 DATA 228,193, 32,228,193, 32,140,193,160,193
150 DATA 169,109, 32, 30,171,169,  0,141, 84,  2
160 DATA 169, 40,133,253,169,216,133,254,169,255
170 DATA 141, 88,  2,169, 40,133,251,169,  4,133
180 DATA 252, 32,  0,193,160,  0,169,  1,145, 45
190 DATA  32,  0,193,145, 45, 32,  0,193,165, 20
200 DATA 145, 45, 32,  0,193,165, 21,145, 45, 24
210 DATA 165, 20,109, 85,  2,133, 20,144,  2,230
220 DATA  21, 32,  0,193,160,  0,140, 86,  2,169
230 DATA 153,145, 45, 32,  0,193,169, 34,145, 45
240 DATA 160,  0,177,251,201, 32,240, 25,201, 96
250 DATA 240, 21,177,253, 41, 15,205, 88,  2,240
260 DATA  12,141, 88,  2,170,189, 93,193, 32,  0
270 DATA 193,145, 45,177,251, 32,  7,193, 32,  0
280 DATA 193,145, 45,230,251,208,  2,230,252,230
290 DATA 253,208,  2,230,254,165,252,201,  7,208
300 DATA   6,165,251,201,232,240, 37,238, 86,  2
310 DATA 173, 86,  2,201, 25,208,179, 32,  0,193
320 DATA 169, 34,160,  0,145, 45, 32,  0,193,169
330 DATA  59,145, 45, 32,  0,193,169,  0,168,145
340 DATA  45, 76, 51,192,160,  0, 32,  0,193,169
350 DATA  34,145, 45, 32,  0,193,169, 59,145, 45
360 DATA  32,  0,193,169,  0,145, 45, 32,  0,193
370 DATA 145, 45, 32,  0,193,145, 45, 32,  0,193
380 DATA  32, 51,165,108,  2,  3,230, 45,208,  2
390 DATA 230, 46, 96, 72, 16, 22,173, 84,  2,208
400 DATA  36,169,  1,141, 84,  2, 32,  0,193,169
410 DATA  18,160,  0,145, 45, 76, 51,190,173, 84
420 DATA   2,240, 14,169,  0,141, 84,  2, 32,  0
430 DATA 193,169,146,160,  0,145, 45,104, 41,127
440 DATA 201, 34,208,  3,169, 39, 96,201, 32,176
450 DATA   4, 24,105, 64, 96,201, 64,176,  1, 96
460 DATA 201, 96,176,  4, 24,105, 32, 96,201,128
470 DATA 176,  4, 24,105, 64, 96,169, 63, 96,144
480 DATA   5, 28,159,156, 30, 31,158,129,149,150
490 DATA 151,152,153,154,155, 19, 18, 85, 84, 73
500 DATA  76, 73, 80, 82, 73, 78, 84,146, 32, 32
510 DATA  32, 32, 32, 13,  0,234,234,234,234,234
520 DATA 234,234,234,234,234,234,160,  0,177, 43
530 DATA 208, 15,200,177, 43,208, 10,173, 85,  2
540 DATA 133, 20,169,  0,133, 21, 96,165, 43,133
550 DATA 251,165, 44,133,252,160,  0,177,251,133
560 DATA 253,200,177,251,133,254,160,  0,177,253
570 DATA 208, 31,200,177,253,208, 26,160,  2,177
580 DATA 251,133, 20,200,177,251,133, 21, 24,165
590 DATA  20,109, 85,  2,133, 20,165, 21,105,  0
600 DATA 133, 21, 96,165,253,133,251,165,254,133
610 DATA 252, 76,169,193, 56,165, 45,233,  1,133
620 DATA  45,165, 46,233,  0,133, 46, 96
```

LOS REGISTROS DEL CHIP TED

Mover la pantalla en una determinada dirección o provocar una interrupción del timer, entre otras funciones, es posible. Sólo basta conocer los bits que constituyen los diferentes registros de la Drean Commodore 16.



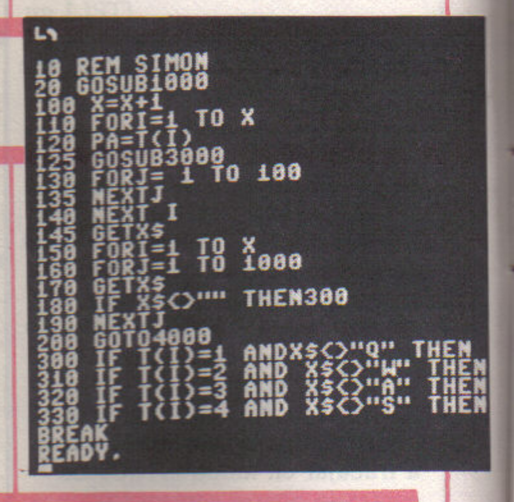
Como recordarán, en el número anterior hemos comentado como se generan las interrupciones para que el sistema pueda barrer el teclado, incrementar el reloj de software (el famoso TI$), etc.
Dijimos que los registros 0,1,2,3,4 y 5 son los encargados de realizar esas funciones del sistema.
Pasemos, pues a la descripción del siguiente registro: el número 6.
Se encuentra en la dirección $FF06-(65286).
Cada uno de los 8 bits que lo constituyen, producen un determinado hecho. Como siempre enumeraremos dichos bits desde el 0 (es más débil) hasta el 7 (el más pesado).
Los bits 0, 1 y 2 determinan la posición del scroll vertical.
Recuerden que "scroll" significa mover toda la pantalla en una determinada dirección (horizontal o vertical).
En este caso el scroll vertical significa mover la pantalla un renglón hacia arriba. De ésta desaparece el primer renglón y aparece, abajo de todo, uno nuevo (como ejemplo, recuerden lo que sucede cuando se ejecuta un LIST).
Originalmente el valor de estos bits es de 011.
El bit 3 se utiliza para indicarle al sistema la cantidad de renglones que contendrá la pantalla. Si está en "0" se representarán 24 filas mientras que con un "1" se representarán 25.
A través del bit 4 se activa y desactiva la pantalla permitiendo, en el segundo caso, que el procesador ejecute las sentencias más rápidamente.
Con un "1" se activa y con un "0" se desactiva.
Los bit 5 y 6 se usan para determinar el modo gráfico. Si el bit 5 está puesto a "1" se activará el modo color.
Finalmente el bit 7 del registro 6 se usa para un cheque del circuito integrado. Siempre debe estar a "0".
El contenido de este registro es, al prenderse la computadora, de 00011011.
El registro 7 ($FF07-65287), al igual que el 6, hace trabajar a cada uno de sus bits. Los bits 0,1 y 2 determinan el scroll horizontal.
El bit 3 determina el modo pantalla en 38 o 40 columnas. Si

está puesto a "0" la pantalla tendrá 38 columnas, mientras que con un "1" estará en 40 columnas.
Luego deberemos incrementar los bits 0,1 y 2 para ocasionar el scroll horizontal, es decir trasladar toda la pantalla hacia la derecha (como en los juegos).
El bit 5 se utiliza para parar el sistema. Esto se logra poniéndolo a "1".
El bit 6 determina el sistema de TV (PAL o NTSC). Si este se pone a "1", se fuerza al sistema a trabajar en modo NTSC. Lo contrario ocurre con un "0" (PAL).
Que se trabaje con NTSC o PAL no significa que podemos trabajar con un TV binorma o algo por el estilo.
Esto se usa, solamente, para tener referencias sobre la frecuencia utilizada para operar.
El bit 7 se utiliza para impresión de caracteres en video inverso.
El registro 8 ($FF08-65288) es utilizado por el teclado.
El registro 9 ($FF09-65289) es el encargado de la administración de las interrupciones del sistema.
El bit 1 de este registro indica al sistema cuando se han imprimido una determinada cantidad de líneas.
El bit 3 señala cuando el timer 1 (el primer reloj) ha llegado a cero. En tanto, el bit 4 y 6, cuando los timers 2 y 3 llegan a cero respectivamente.
El bit 2 indica una interrupción proveniente del lápiz óptico.
Ese bit se utiliza para futuras expansiones.
El sistema pone un "1" en el bit correspondiente indicando que se ha llegado a una condición de interrupción.
Por ejemplo si el bit 3 está a "1", indicará que el timer 1 ha llegado a cero.
El registro 10($FF0A-65290) determina qué fuente (es decir timer1, timer2, timer3, o barrido) ocasionará interrupción.
Si ponemos un "1" en, por ejemplo, el bit 1 de este registro, permitiremos que se provoque una interrupción cuando el timer 1 llegue a cero.
En cambio, si ponemos un "1" en, por ejemplo, el bit 1 de este registro, permitiremos que se provoque una interrupción cuando el timer 1 llegue a cero.
En cambio, si ponemos un "1" en el bit 6, permitiremos también que el timer 3 ocasione una interrupción IRQ.
Con un "0" no se dejará que el timer 3 provoque interrupción.
El registro 11 ($FF0B-65291) es utilizado como contador. Su función es contar la cantidad de líneas que se imprimen en la pantalla.
No se deben confundir con las líneas de impresión normal de la Drean Commodore 16 (0—24).
Estas líneas corresponden al barrido que se realiza sobre el TV para poder imprimir los caracteres.

SIMON

Tipo: Juego
Comp.: DC-64
Conf.: Básica

Se acuerdan del famosísimo Simon, aquel juego cuyo fin era el de repetir los tonos generados por la "maquinita"?
Si es así, aquí les damos la versión para la Drean Commodore 64.
Para aquellos que no se acuerden, le decimos que el objetivo es repetir, como dijimos antes, los tonitos musicales generados por la computadora. Para ello se utilizan cuatro teclas: la A, S, Q y W.
Cada una de ellas se irá mostrando en la pantalla.
Nosotros debemos seguir la secuencia evitando equivocarnos.
Si ello ocurre, un espantoso tono nos indicará de tal hecho.
Sólo se necesita un requisito para ser un triunfador en los dominios del Simon: ¡concentración!.



VARIABLES UTILIZADAS

X:	Contador puntaje.
I,J:	Indice
PA:	Copia temporal del tono generado al azar.
T(I):	Contiene los tonos generados.
X$:	Contiene a la tecla oprimida.
V:	Chip VIC-II dirección de inicio de todos sus registros.
HF:	Frecuencia alta
LF:	Frecuencia baja
W:	Ancho del pulso
PX:	Posición eje X del sprite
PY:	Posición eje Y del sprite
RE:	Puntuación máxima.

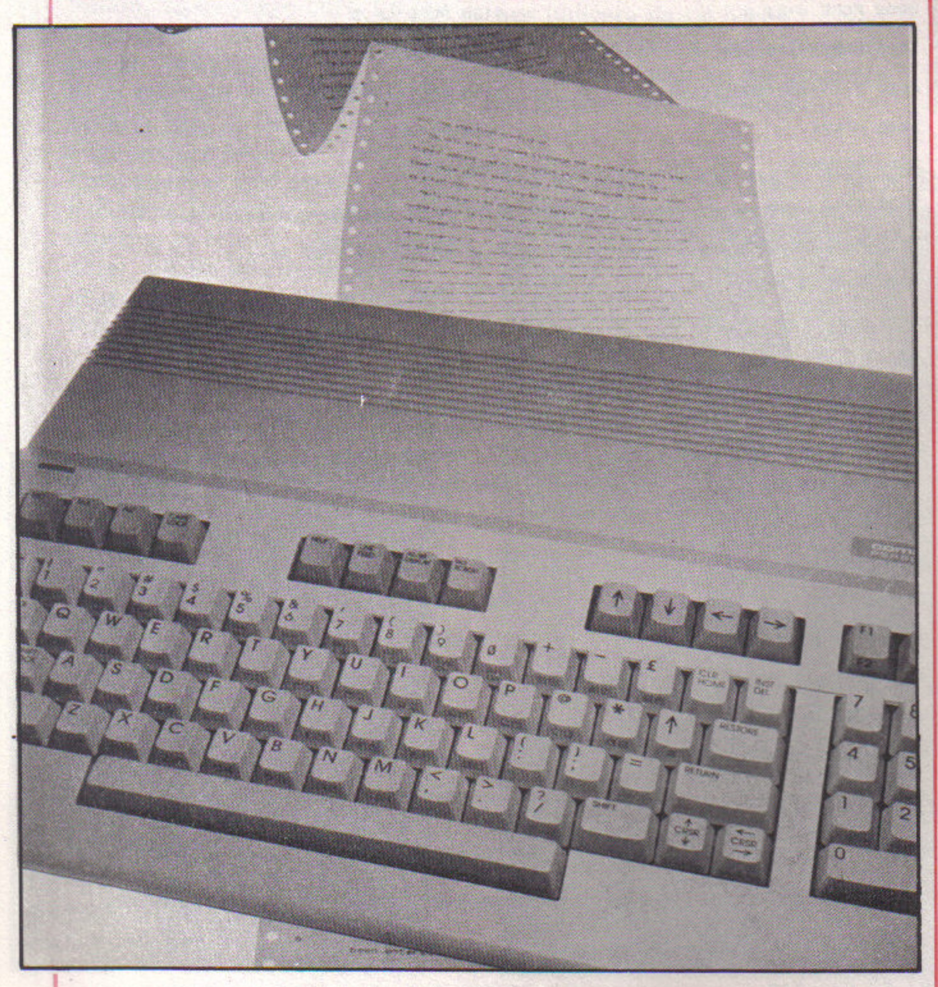
DESCRIPCION DEL PROGRAMA

Línea	Descripción
1000	Dimensiona el vector T a 1000 elementos.
1005	Borra la pantalla.
1010	Asigna a V la dirección de inicio del controlador del video VIC-II
1020-1070	Pone valor inicial del primer tono.
1080-1100	Determina frecuencia alta, baja y pulso del sonido
1110-1120	Determina posición del primer sprite.
1130-1200	Determina posición de los cuatros sprite que representan a las letras Q,W,S y A.
1250-1320	Expande los cuatro primeros sprites (0,1,2 y 3) en X e Y. Determina el color de cada uno de ellos. Determina color de borde y color de fondo.
1500-1530	Pone formas de los sprites en el área correspondiente al buffer del cassette.
2000-2040	Apaga todos los sprites y comienza generación al azar de los distintos tonos.
3000-3100	Genera tono y posición del Simon.
4000-4310	Genera tono de bajo cuando no se oprime la tecla correspondiente. Imprime puntos obtenidos y pregunta por otro juego.
1000-1360	Forma de los cuatro sprites del Simon.

```
10 REM SIMON
20 GOSUB1000
100 X=X+1
110 FORI=1 TO X
120 PA=T(I)
125 GOSUB3000
130 FORJ= 1 TO 100
135 NEXTJ
140 NEXT I
145 GETX$
150 FORI=1 TO X
160 FORJ=1 TO 1000
170 GETX$
180 IF X$<>"" THEN300
190 NEXTJ
200 GOTO4000
300 IF T(I)=1 AND X$<>"Q" THEN 4000
310 IF T(I)=2 AND X$<>"W" THEN 4000
320 IF T(I)=3 AND X$<>"A" THEN 4000
330 IF T(I)=4 AND X$<>"S" THEN 4000
340 PA=T(I)
350 GOSUB3000
360 NEXTI
370 FORI=1 TO 500
380 NEXTI
400 GOTO100
1000 DIM T(100)
1005 PRINT CHR$(147);
```

```
1010 V=53248
1015 X=0
1020 POKE 54296,15
1040 POKE 54277,190
1050 POKE 54275,15
1060 POKE 54274,15
1070 POKE 54278,136
1080 HF=54273
1090 LF=54272
1100 W=54276
1110 PX=135
1120 PY=120
1130 POKE V,PX
1140 POKE V+1,PY
1150 POKE V+2,PX+38
1160 POKE V+3,PY
1170 POKE V+4,PX
1180 POKE V+5,PY+44
1190 POKE V+6,PX+38
1200 POKE V+7,PY+44
1210 POKE 2040,13
1220 POKE 2041,14
1230 POKE 2042,15
1240 POKE 2043,16
1250 POKE V+29,15
1260 POKE V+23,15
1270 POKE V+39,5
1280 POKE V+40,6
1290 POKE V+41,7
1300 POKE V+42,2
1310 POKE 53280,1
1320 POKE 53281,1
1330 H=8
1340 L=42
1400 RESTORE
1500 FORI=0 TO 255
1510 READ J
1520 POKE 832+I,J
1530 NEXTI
2000 POKE V+21,0
2020 FORI= 1TO 100
2030 T(I)=INT(RND(TI)*4)+1
2040 NEXTI
2050 RETURN
3000 POKE W,65
3010 POKE HF,H*PA
3020 POKE LF,L*PA
3030 POKE V+21,2↑(PA-1)
3040 FORJ=1 TO 50
3050 NEXTJ
3060 POKE V+21,0
3070 POKE HF,0
3080 POKE LF,0
3090 POKE W,0
3100 RETURN
4000 POKE HF,4
4010 POKE LF,73
4020 POKE W,65
4030 PRINT CHR$(144)
4040 FORI=1 TO 5
4050 PRINT
4060 NEXTI
4070 PRINTTAB(10)"ESA NO FUE
LA LETRA !!!":PRINTTAB(10)"UD
PERDIO"
4080 FORI=1 TO 500
4090 GETX$
4100 NEXTI
4110 POKE HF,0
4120 POKE LF,0
4130 POKE W,0
4140 FORI=1 TO 4
4150 PRINT
4160 NEXTI
4170 PRINTTAB(13)"PUNTOS :";X-1
4180 IF RE<X-1 THEN RE=X-1
4190 PRINT
4200 PRINT
4210 PRINT
4220 PRINTTAB(10)"PUNTUACION
     MAXIMA :";RE
4230 PRINT
4240 PRINT
4250 PRINT
4260 PRINT TAB(13)"OTRO JUEGO ?"
4270 GETX$
4280 IF X$="" THEN4270
4290 IF X$="N" THEN END
4300 GOSUB 1005
4310 GOTO100
10000 DATA  0,  0,248,  0,  7,248,  0, 31,248
10010 DATA  0,127,248,  1,255,248,  3,255,248
10020 DATA  7,255,248, 15,255,248, 15,227,248
10030 DATA 31,221,248, 31,221,248, 63,221,248
10040 DATA 63,221,248,127,217,248,127,225,248
10050 DATA127,254,248,255,255,248,255,255,248
10060 DATA255,255,248,255,255,248,255,255,248,  0
10100 DATA 31,  0,  0, 31,224,  0, 31,248,  0
10110 DATA 31,254,  0, 31,255,128, 31,255,192
10120 DATA 31,255,224, 31,255,240, 31,187,240
10130 DATA 31,187,248, 31,187,248, 31,171,252
10140 DATA 31,171,252, 31,171,254, 31,215,254
10150 DATA 31,255,254, 31,255,255, 31,255,255
10160 DATA 31,255,255, 31,255,255, 31,255,255,  0
10200 DATA255,255,248,255,255,248,255,255,248
10210 DATA255,255,248,255,255,248,127,255,248
10215 DATA127,227,248,127,221,248, 63,221,248
10220 DATA 63,221,248,. 31,193,248, 31,221,248
10230 DATA 15,221,248, 15,255,248,  7,255,248
10240 DATA  3,255,248,  1,255,248,  0,127,248
10250 DATA  0, 31,248,  0,  7,248,  0,  0,248,  0
10300 DATA 31,255,255  31,255,255, 31,255,255
10310 DATA 31,255,255, 31,255,255, 31,255,254
10320 DATA 31,199,254, 31,187,254, 31,191,252
10330 DATA 31,199,252, 31,251,248, 31,187,248
10340 DATA 31,199,240, 31,255,240, 31,255,224
10350 DATA 31,255,192, 31,255,128, 31,254,  0
10360 DATA 31,248,  0, 31,224,  0, 31,  0,  0,  0
```

SECRETOS DE LA PROGRAMACION ESTRUCTURADA



Gracias al nuevo Basic V7.0 de la Commodore 128, disponemos de herramientas sumamente potentes orientadas a la programación estructurada.
En esta nota les explicamos cómo utilizar a fondo esas ventajas.

Casi todos coinciden en que la programación estructurada es una técnica de diseño y documentación de programas que impone una estructura uniforme a todos ellos.
Por eso es común oir que ciertos lenguajes son estructurados y otros que no lo son.
Por ejemplo, el Pascal es un lenguaje estructurado y el Basic no. Esto se debe al diseño del lenguaje en sí.
Siguiendo en el caso de Pascal, N. Wirth (su creador) lo hizo teniendo en cuenta, entre otros aspectos, el diseño de un programa en Pascal.
El resultado de esta premisa es que, entre otras reglas, el programador está totalmente imposibilitado de usar variables que no hayan sido definidas al comienzo de su programa.
De esta manera él, antes de ponerse a hacer software, realiza una estructura del programa.
Una de las principales reglas que se imponen dentro de la programación estructurada es que cada módulo sólo tiene una entrada y una única salida.

Listado 1

```
10 INPUT"INGRESE UN NUMERO";N
20 IF N<100 THEN 100
30 continua el programa
```

Listado 2

```
10 DO
20 : INPUT"INGRESE UN NUMERO";N
30 LOOP UNTIL N>100
40 continua el programa
```

Esto quiere decir que las sentencias GOTO no se pueden utilizar.
El programador Basic se preguntará cómo se hacen los programas si no se cuenta con esta "herramienta" tan frecuentemente utilizada por ellos.

Sin embargo, así como se quitan algunas cosas, se agregan otras que son, sobre todo, muchísimo más potentes.

Esas cosas se conocen con el nombre de estructuras. Por ejemplo, el lazo FOR-NEXT es una estructura de repetición. Existen otras, para la Commodore 128, que son UNTIL y WHILE.

Como dijimos antes, la Commodore 128 dispone, dentro de su nuevo Basic 7.0, de muy buenas y potentes sentencias. Nosotros sólo nos encargaremos de mostrar las sentencias UNTIL, WHILE y IF-BEGIN-BEND.

Listado 3

```
10 INPUT"INGRESE UN NUMERO";N$
20 I=1
30 DO
40 : A$=MID$(N$,I,1)
50 : A=ASC(A$)-48
60 : IFA$="F"THENA=15
70 : IFA$="E"THENA=14
80 : IFA$="D"THENA=13
90 : IFA$="C"THENA=12
100 : IFA$="B"THENA=11
110 : IFA$="A"THENA=10
120 : N=N+A*16↑(LEN(A$)-I)
130 : I=I+1
140 LOOP UNTIL I>LEN(A$)
150 PRINTN
```

Listado 4

```
10 DO WHILE A$=""
20 :  GETA$
30 LOOP
```

Figura 1

```
10 DO
20 :  sentencia 1
30 :  sentencia 2
40 :  sentencia 3
50 :  .......... .
60 :  sentencia n
70 LOOP UNTIL (predicado)
```

UNTIL es una estructura de repetición que permite ejecutar un determinado módulo hasta que se cumpla una cierta condición.
Como módulo se entiende un grupo finito de sentencias encerradas entre un DO y un LOOP UNTIL (siempre dentro del caso de la C-128).
La figura 1 representa esta estructura. La primera sentencia es un DO. Luego le siguen las sentencias que constituyen al módulo. Finalmente lo cerramos a través del LOOP UNTIL junto con la condición (predicado) que se debe verificar para que se salga del lazo.

Listado 5

```
10 INPUT"INGRESE STRING";A$
20 I=1:C=0
30 DO WHILE MID$(A$,I,1)<>"."
40 :     IF MID$(A$,I,1)="A"THEN C=C+1
50 :     I=I+1
60 LOOP
70 PRINT"CANTIDAD DE A:";C
80 END
```

En caso de que dicho predicado sea falso (es decir no se ha cumplido la condición) se volverá a repetir dicho módulo.

La figura 2 representa el cumplimiento de la "regla de oro". Nuestro bloque verifica que tiene una entrada y una sola salida.

Supongamos validar una entrada, la cual debe ser un número entero mayor que 100. El programa debe "rebotar" todos aquellos números menores o iguales a 100.
El listado 1 realiza esta tarea usando los comandos comunes. El listado 2 usa la sentencia UNTIL.
En este caso, mientras se ingresen números menores o iguales a 100 se volverán a ejecutar todas las líneas comprendidas entre DO y LOOP.
Otras de las ventajas que surgen gracias a la utilización de estas nuevas sentencias, consiste en la claridad de los programas.
Para que esto se comprenda mejor, veamos otro segundo ejemplo.
Hagamos un programa que realice una conversión numérica (de hexadecimal a decimal).
Básicamente el objetivo de éste es hacer notar como aumenta la claridad del programa y por ende su posterior mantenimiento.
El programa corresponde al listado 3.
La sentencia WHILE es, también, una estructura de repetición pero su predicado se encuentra al principio del bloque. A diferencia de UNTIL en donde, por lo menos, el conjunto de instrucciones se ejecuta una vez debido a que el predicado evaluativo (condición) se

Listado 6

```
10 INPUT"INGRESE UN NUMERO";N
20 IFN/2=INT(N/2)THENBEGIN:PRINT"EL NUMERO ES PAR"
30 BEND:ELSE:PRINT"EL NUMERO ES IMPAR"
```

encuentra al final de esas sentencias, la estructura del WHILE evalúa la condición antes de iniciar la ejecución del bloque.

De esta manera el bloque no se ejecutará si la condición es cierta. Se omitirán todas las sentencias ubicadas entre el WHILE y el LOOP.
La figura 3 representa a la estructura DO WHILE.
Un ejemplo sencillo utilizando esta sentencia se representa en el listado 4. Aquí se lee el teclado hasta que se oprima una tecla.
Estamos diciendo, usando el WHILE, que mientras A$ sea igual a " " (es decir no se oprima ninguna tecla) se sigue leyendo el teclado.
En caso contrario se continuará con el programa.
Un ejemplo un tanto más complicado se indica en el listado 5. Este representa la resolución del siguiente problema: Se ingresa una cadena (string) de caracteres que termina con ".".
Se pide contar la cantidad de letras "A" que ese string contiene.
En la línea 10 leemos el string asignándolo a la variable A$.
En la línea 20 inicializamos el puntero que se moverá por el string a 1 y el contador de "aes" a 0.

En la línea 30 comenzamos la ejecución del módulo principal, que se ejecutará mientras el caracter leído sea distinto del "." (fin de texto).
Sólo finalizaremos la ejecución del mismo cuando se llegue a la condición de fin de procesamiento.
En este caso el control se transferirá a la línea 70 en donde se imprimirá la cantidad de "aes".
Otra de las herramientas utilizadas en programación estructurada es el bloque de decisión que permite ejecutar sus bloques acorde a la falsedad o veracidad de la condición.
Nos estamos refiriendo al IF-THENBEGIN-BEND:ELSE
En la figura 4 se muestra dicha estructura. Cada bloque representa una o varias sentencias.
La Commodore 128 utiliza dos palabras reservadas para indicar el comienzo y el fin del bloque "verdadero"; es decir aquél se ejecutará sólo si la condición es verdadera.
Esas palabras son BEGIN (comienzo) y BEND (fin). Con otro ejemplo se verá mejor el funcionamiento de este "especial" IF-THEN.
El listado 6 es un programa que determina si un número es par o impar. Primero se toma el número. Luego se comprueba si el número es par.
En caso afirmativo se procede a informar de este hecho. En caso negativo, el programa saltea las líneas del programa hasta el BEND y ejecuta de ahí en adelante.
Por último existe otro tipo de estructura de repetición que permite ejecutar indefinidamente una serie de sentencias.
Esta es el lazo DO-LOOP que equivale a la sentencia GOTO siempre en la misma línea.
Los listados 7 y 8 son ejemplos de cada uno de ellos.
En futuras notas continuaremos explicando nuevas técnicas de programación junto con sus correspondientes estructuras de datos.

Figura 2

ENTRADA (DO)
(Sentencia 1)
(Sentencia 2)
•
•
•
(Sentencia h)
SALIDA (LOOP UNTIL)

Figura 3

```
10 DO WHILE(predicado)
20 :  sentencia 1
30 :  sentencia 2
40 :  sentencia 3
50 :  ......... .
60 :  sentencia n
70 LOOP
```

Figura 4

ENTRADA
CONDICIÓN
V | F
BLOQUE 1 | BLOQUE 2
SALIDA

Listado 7

```
10 DO
20 : sentencia 1
30 : sentencia 2
40 : ......... .
50 : sentencia n
60 LOOP
```

Listado 8

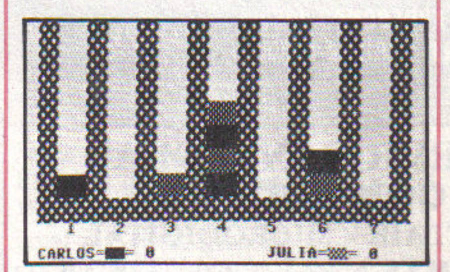
```
10 sentencia 1
20 sentencia 2
30 ......... .
40 sentencia n
50 GOTO 10
```

GUIA PRACTICA GUIA PRACTICA

SUPERCUAD

Esperamos que con SUPERCUAD agudicen su ingenio tratando de ganarle a sus contrincantes.

Tipo: Juego
Comp.: DC-64
Conf.: Básica



El objetivo de este programa es que uno de los jugadores que participan pueda alinear horizontal, diagonal o verticalmente cada uno de los cuadrados que maneja.
Al comienzo se nos preguntará si deseamos instrucciones. Aquí debemos tipear 'S' o 'N'.
En caso de oprimir 'N' se nos interrogará sobre el nombre del primer jugador. Luego el del segundo.
Finalmente aparecerán en la pantalla siete columnas.
Comienza el jugador 1 tipeando el número de columna donde quiere que su pieza caiga. Lo sigue el jugador 2. Así el ciclo se repite hasta que haya un ganador o hasta que el juego se empate.

DESCRIPCION DEL PROGRAMA

Linea	Descripción
10	Determina color de pantalla y de fondo.
100	Asigna caracteres de control del cursor.
120	Dimensiona los vectores utilizados.
160-190	Toma el nombre de los dos jugadores. En caso de que superen los seis caracteres se queda con los seis primeros.
200-240	Imprime tablero de juego.
300-390	Toma desde el teclado la posición de las piezas.
400-560	Imprime piezas determinando si hay ganador.
570-690	Indica que jugador fue el ganador, incrementa el score de ese jugador en uno y reinicializa todos los valores y gráficos para la siguiente partida.

VARIABLES UTILIZADAS

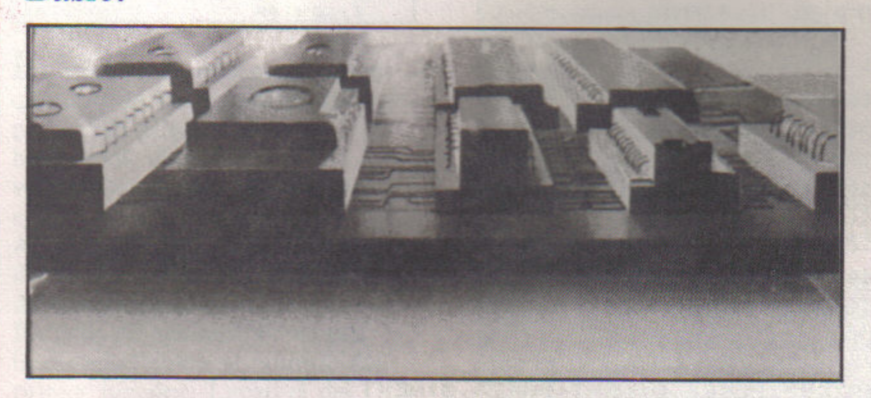
ROM:	Representa la dirección de memoria que indica la cantidad de caracteres en el buffer del teclado
GÓ$:	Variable de control del cursor.
N$(I):	Vector que indica los nombres de los jugadores.
U$:	Variable auxiliar
J:	Variable indice.
L,Q,Z:	Auxiliares
R$:	Toma datos del teclado. Indica columna elegida para el movimiento
SQ:	Contador parcial
A%,B%:	Vectores de posición de movida

```
10 ROM=198:POKE53280,1:POKE53281,1
100 GO$="████████████████████████ "
120 DIMA(7),A%(7),B(7),B%(8,8),N$(2):SQ=49:OPEN1,0,0:M$="████"
130 PRINTM$"NECESITA INSTRUCCIONES   ? "
140 GETU$:IFU$=""THEN140
150 IFU$<>"S"ANDU$<>"N"THEN140
160 IFU$="S"THEN700
170 FORJ=1TO2:PRINTM$"JUGADOR"J"(MAX.=6 LETRAS) ? ";:INPUT#1,N$(J)
180 IFLEN(N$(J))>6THENN$(J)=MID$(N$(J),1,6)
190 NEXT
200 PRINT"████":FORZ=1TO15
210 PRINT" XX    XX    XX    XX    XX    XX    XX    XX"
220 NEXT
230 FORZ=1TO2:PRINT" XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX":NEXT
240 PRINT"█    1    2    3    4    5    6    7██"
250 FORF=1TO7:READA(F):NEXT
260 DATA1187,1267,1347,1427,1507,1587,1667
```

```
270 FORF=1TO7:READB(F):NEXT
280 DATA1027,1032,1037,1042,1047,1052,1057
290 PRINTGG$;N$(1)"=■  ■="WH;"                  "N$(2)"=■■="BL
300 GETR$:IFR$=""THEN300
310 IFVAL(R$)=0ORVAL(R$)>7THEN300
320 R=VAL(R$)
330 IFA%(R)=7THEN300
340 IFN=160THENN=102:POKE1989,81:POKE56261,0:POKE2012,32:GOTO360
350 N=160:POKE2012,81:POKE56284,0:POKE1989,32
360 GOSUB400  :A%(R)=A%(R)+1:GOTO430
370 SQ=SQ-1:POKEROM,0
380 IFSQ=0THENPRINT"■"SPC(15)"EMPATE...!!":FORBB=1TO1500:NEXT:GOTO640
390 GOTO300
400 FORW=B(R)TOA(7-A%(R))+5*RSTEP40
410 FORQ=WTOW+2:POKEQ+40,N:POKEQ-40,32:NEXT:POKEA,N/2:FORT=1TO5:NEXT:POKEA,0
420 NEXT:RETURN
430 B%(R,A%(R))=N
440 FORH=A%(R)TO1STEP-1:IFB%(R,H)<>NTHEN460
450 CO=CO+1:NEXT
460 IFCO>=4THEN570
470 FORH=1TO3:IFB%(R+H,A%(R))=NTHENNEXT
480 FORI=1TO3:IFB%((R-I),A%(R))=NTHENNEXT
490 IFI+H>4THEN570
500 FORH=1TO3:IFB%(R+H,A%(R)+H)=NTHENNEXT
510 FORI=1TO3:IFB%(R-I,A%(R)-I)=NTHENNEXT
520 IFI+H>4THEN570
530 FORH=1TO3:IFB%(R-H,A%(R)+H)=NTHENNEXT
540 FORI=1TO3:IFB%(R+I,A%(R)-I)=NTHENNEXT
550 IFI+H>4THEN570
560 CO=0:GOTO370
570 FORL=1TO3:FORQ=0TO255STEP5:POKEA,Q:NEXT:POKEA,0:NEXT
580 FORW=1074TO1114STEP40
590 FORQ=WTOW+2:POKEQ,N:NEXT:NEXT
600 PRINT"■■■"SPC(14)"ES EL GANADOR !!!"
610 FORRR=1TO1000:NEXT
620 IFN=102THENBL=BL+1:GOTO640
630 WH=WH+1
640 RESTORE:FORXX=1TO7:A%(XX)=0:NEXT:SQ=49:CO=0
650 FORXX=1TO7:FORXV=1TO7:B%(XX,XV)=0:NEXT:NEXT
660 PRINT"■";:FORTV=1TO3:PRINT"                                        ";:NEXT
670 PRINT:FORZ=1TO15
680 PRINT"■ XX    XX    XX    XX    XX    XX    XX    XX■"
690 NEXT:POKEROM,0:N=102:POKE1989,81:POKE2012,32:GOTO250
700 PRINT"■■    ";
710 FOREE=1TO22:PRINT"SUPERCUAD    ■■■■■■■■■■■■■■■";:NEXT
720 FORT=1TO2000:NEXT:PRINT"■"
730 PRINT" EL OBJETIVO DEL JUEGO ES ALINEAR CUATRO DE SUS PIEZAS EN UNA";
735 PRINT" COLUMNA. TAMBIEN   SE PUEDEN ALINEAR DIAGONALMENTE Y"
737 PRINT" HORIZONTALMENTE.";
740 PRINT" COMPITEN DOS JUGADORES Y GANA EL PRIMERO EN UBICAR SUS PIEZAS."
750 PRINT" PRIMERO COMIENZA EL JUGADOR NUMERO 1,   DONDE DEBE ELEGIR LA"
760 PRINT" COLUMNA, LAS QUE ESTAN ENUMERADAS DE 1  A 7"
770 PRINT" LUEGO LE SIGUE EL JUGADOR 2 QUIEN        DEBERA ELEGIR SU COLUMNA."
780 PRINT" EL PROCESO SE REPITE HASTA QUE HALLA UN GANADOR O HASTA QUE";
790 PRINT" EL JUEGO SE EMPATE"
810 PRINTSPC(10)"■■PRIMA UNA TECLA !"
820 GETU$:IFU$<>""THEN170
830 GOTO820
```

LAS SUBRUTINAS DEL DREAN COMMODORE 64

Establecer el inicio o el final de la memoria libre, o realizar la apertura de un archivo lógico son,entre otros, propósitos de los usuarios. Veamos como lograrlo desde el Assembler o el Basic.



Nombre de la función: MEMBOT
Propósito: Establece el inicio de la memoria libre.
Dirección de llamada: $FFC8 (hex) 65436 6DEC7
Registros de comunicación:X,Y
Rutina preliminar: Ninguna
Error: Ninguno
Requerimientos de stack: ninguno
Registro afectados: X,Y
Descripción: Esta rutina se utiliza para determinar el inicio de la memoria libre (en donde se almacenan nuestros programas).
En realidad esta rutina también se usa para establecer el inicio de esa memoria. El sistema operativo resetea o lee el inicio de la memoria de acuerdo con el flag Carry.
Si éste se encuentra en "1" antes de acceder a la rutina, se devuelve en los registros X e Y la parte baja y alta de la dirección inicial del área libre.
En cambio si está a "O", el inicio de la memoria libre será a partir de la dirección determinada por los registros X e Y (parte baja y alta respectivamente).
Pasos a seguir: 1) Para leer el inicio de la memoria se debe poner el flag de Carry a "1" y luego acceder a la rutina.

2) Para establecer el nuevo inicio se debe poner el flag de Carry a "0", poner en los registros X e Y la nueva dirección y luego acceder a la rutina.

Ejemplo:

a) Desde el Assembler

```
SEC; ponemos Carry en "1" para
      leer inicio
JSR $FFC9; accedemos a la rutina.
```

b) Desde el Basic:

```
10POKE783,PEEK(783)OR1:REM
PONEMOS CARRY EN "1'
20 SYS65436: REM ACCEDEMOS
A LA RUTINA
30 PRINTPEEK(781)+256*PEEK
  (782)
:REM IMPRIMIMOS INICIO DE
MEMORIA LIBRE
40 REM 781 Y 782
REPRESENTAN A LOS
REGISTROS X E Y
50 REM 783 REPRESENTA AL
REGISTRO DE ESTADO DEL
MICROPROCESADOR
60 END
```

Nombre de la función: MEMTOP
Propósito: Establece el final de la memoria libre.
Dirección de llamada: $FF99 (hex) 65433 (dec)
Registros de comunicación:X,Y
Rutina preliminar: Ninguna
Error: Ninguno
Requerimientos de stack:2
Registros afectados:X,Y
Descripción: En contraposición con la rutina anterior, ésta rutina se utiliza para determinar el final de la memoria libre.
Igualmente es posible leer o reestablecer el final del área libre. El procedimiento para conocer el fin de dicha área es igual que el anterior.

Para ello se debe poner en "1" el flag de Carry y luego acceder a esta rutina. En los registros X e Y se devolverán la parte baja y alta de la última dirección de esta área.
Si el acceso a esta rutina se realiza con el Carry a "0", el fin de la memoria libre quedará determinada por la dirección representada en los registros X e Y (parte baja y alta respectivamente).
Pasos a seguir:

1) De acuerdo a si se va a leer o a poner una nueva dirección de fin de área libre, poner el Carry a "1" o "0"
2) Acceder a la rutina.

Ejemplo:

a) Desde el Assembler

```
SEC; ponemos Carry en "1" para
      leer fin
JSR $FF99; accedemos a la rutina
```

b) Desdel el Basic:

```
10 POKE783,PEEK(783)OR1:REM
PONEMOS CARRY EN "1"
20 SYS65433:REM ACCEDEMOS A
LA RUTINA
30 PRINTPEEK(781)+256*PEEK
  (78
2):REM IMPRIMIMOS FIN DE
MEMORIA LIBRE
40 END
```

BASIC Y ACELERADOR DE DISCO PARA LA DREAN COMMODORE 64

Nuevos comandos para programas, expandir la memoria o aumentar la velocidad de transferencia del disco. Ahora es más factible gracias a los desarrollos que posibilita la industria nacional.



Hal S.A. ha desarrollado cuatro nuevos productos para la Drean Commodore 64 y C-128. Se trata del HAL BASIC, HAL EXPANDER, HAL CHARGET y el HAL M-128, todos ellos disponibles en cartridge con su correspondiente botón de reset.
El primero de ellos es un Basic mejorado, el cual suministra al programador un total de 114 nuevos comandos.
Por ejemplo, existen comandos orientados al manejo de sonido y gráficos. De esta manera resulta más fácil realizar composiciones musicales y gráficas.

También se suman al Basic residente sentencias orientadas a la programación estructurada como por ejemplo DO-LOOP.

Con respecto al manejo de errores, el HAL BASIC dispone del comando ON ERROR, que transfiere el control del programa a una determinada línea, en caso de producirse un error.

También se suministran comandos como el KEY, RENUMBER, AUTO y PAUSE. Lo acompaña el manual correspondiente con más de 80 páginas.

El HAL EXPANDER agranda la memoria de la Drean Commodore en 22 Kbytes. De esta manera, el programador dispone de 61183 Kbytes.

Además suministra nuevos comandos como el PRINT USING permitiendo la salida en pantalla o en impresora en formatos predefinidos.

Contiene una sentencia HELP la cual hace imprimir en pantalla la línea del programa en donde se ha cometido un error (sintaxis, bad subscript, etc.).
En total son 50 los nuevos comandos. Junto a su manual está el mapa de memoria de la Drean Commodore 64.
El CHARGER permite aumentar la velocidad de transferencia del disco hacia o desde la computadora más de 10 veces de lo normal.
Incluye un editor de Assembler, copiador y reset.
Por último, el HAL M-128 es uno de los productos cuya finalidad es aumentar la velocidad de la 1541 o de la 1571.
Es decir que este cartucho puede trabajar con la Drean Commodore 64 o con la Commodore 128.
También le permite al programador realizar conversiones numéricas a través de nuevos comandos.
Amplía la memoria en 4 Kb y formatea un disco en 10 segundos.

Cada uno de los productos comentados son cien por cien nacionales. Todos disponen de un reset incorporado que nos permite reinicializar, en cualquier momento, la Drean Commodore 64 o C-128.

Acompañan a cada producto la correspondiente garantía válida por un año.

DEFINICION DE VARIABLES

Las variables son imprescindibles para programar. Eso todos lo saben. Lo importante es definirlas desde un principio y aquí tratamos de explicar cómo y porqué.



Aquellos que programan "fuerte" y, también, los que recien comienzan a programar, seguramiente nunca han escrito en papel las variables que utilizará el programa.
Simplemente se han limitado a crearlas en el momento de utilizarlas. Evidentemente, ésta no es la mejor manera de programar.

Para que nadie se quede en el camino contestaremos la siguiente pregunta: ¿Qué es una variable?

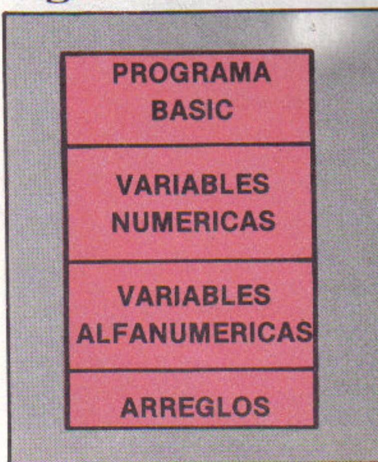
Sin entrar en definiciones de diccionario, una variable es una "cajita" identificada con un nombre y que se encuentra en la memoria de la computadora.

Esa cajita puede tomar determinados valores y que, a lo largo del programa, pueden modificarse.

Por ejemplo, si nuestra cajita contendrá datos numéricos, sólo podemos poner dentro de ella números.
Esto se debe a cómo se representan internamente los valores en la memoria.
En cambio, si nuestra cajita tuviera que contener caracteres, se nos permite poner números dentro de ellas. Al final de cuenta los números también son caracteres.
El nombre es necesario para poder usar más de una caja.
Junto a él se pone o no un caracter especial que indica el tipo de dato que se almacenerá.
De esta manera el signo "$" especifica caja alfanumérica (letras + números). Si se omite se entenderá que es una variable estrictamente numérica.
Por ejemplo, PEPE y PEPE$ es una variable numérica y alfanumérica respectivamente.
Mucho "ojito", sí podemos hacer la operación PEPE/21 pero no PEPE$/21 (¡no tiene sentido dividir por 21 una serie de caracteres!).
Resumiendo; una variable se puede pensar como una cajita a la cual se la identifica con un nombre y que puede almacenar determinados tipos de datos.
Ahora bien, como toda buena caja, ésta ocupa un cierto lugar.
Para nuestro caso este lugar es una determinada sección de memoria.
Es decir que cuando nosotros creamos una variable (ponemos un dato en una cajita) el sistema operativo hace espacio para que ella pueda entrar fácilmente.
Los equipos Drean Commodore tienen determinadas zonas reservadas para el almacenamiento de variables.
La zona de memoria en donde se almacenan las variables numéricas comienza a partir del final de nuestro programa.
Las variables alfanuméricas se almacenan a partir del final del área anterior. Finalmente los arreglos (vectores, matrices, etc)

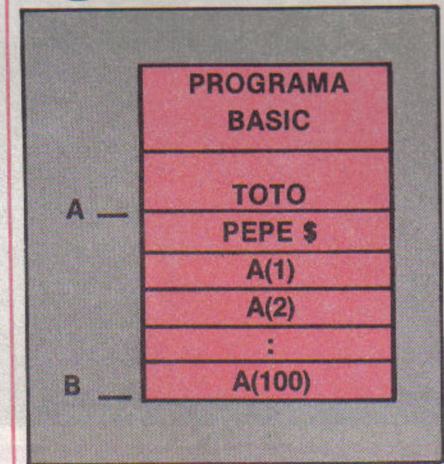
se almacenan a partir del final de esa área.
La figura 1 ejemplifica lo dicho.
Para que se comprenda mejor por qué es útil definir nuestras cajitas al principio, haremos un ejemplo.
Supongamos que nuestro programa usa las variables TOTO, PEPE$ y el vector A conteniendo 100 elementos.

Figura 1



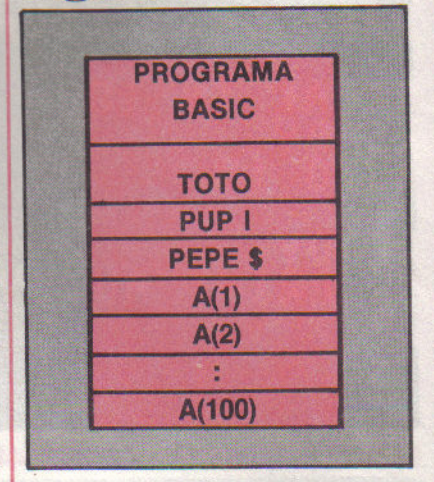
Si esto es así la memoria estará distribuida como lo muestra la figura 2.
En el medio de nuestro programa creamos la variable PUPI. ¿Qué es lo que hace el sistema ante este hecho?
Como PUPI es una variable numérica debe poner su cajita con las demás cajas numéricas. Para ello debe mover toda el área de memoria comprendida entre A y B tantos lugares como se necesiten para que pueda entrar la nueva caja.
Luego de terminar el proceso la memoria queda distribuída como se indica en la figura 3.
El proceso de trasladar toda esa sección no se realiza en forma instantánea sino que, por el contrario, lleva un determinado tiempo.

Figura 2



Obviamente, a medida que el área a trasladar crezca, el tiempo que se tardará para moverla también aumentará.
Y, además, mientras se lleva a cabo ese traslado el programa no se ejecuta, produciéndose la mal vista demora y nuestros nerviosos suspiros.
¿Qué es lo que podemos hacer para evitarlo?
La respuesta es definir las variables al principio del programa. Por ejemplo podemos darles valores de cero a las cajitas numéricas y espacios en blanco a las alfanuméricas.
Esto se traduce en Basic a:
10 TOTO=0:PUPI=0:PEPE$=""
Con respecto al vector debemos respetar el DIM A (100), con lo que le diremos al sistema que reserve lugar para almacenar 100

Figura 3



futuros elementos numéricos.
La segunda ventaja que resulta es que, en cierta forma, estamos "pensando" en las variables a utilizar. Así evitaremos utilizar "cajitas" innecesarias.
Nunca falta el que por no ponerse a pensar sólo cinco minutos, usa más de 7000 variables para hacer una suma entre dos números ¡Nos lo van a decir a nosotros, pssss...!

GENERADOR DE NUMEROS DE LINEA

```
 CARGANDO VALORES. UN MOMENTO...
AUTO ACTIVADO. PARA ACTIVARLO TIPEE AUTO
 SEGUIDO DEL INCREMENTO
PARA DESACTIVARLO SOLO TIPEE AUTO

BREAK IN 180
READY.
AUTO 10

READY.
10 PRINT"AUTO"
20
```

Tipo: Utilitario
Comp.: DC-64
Conf.: Básica

Cuando debemos programar con la Drean Commodore 64 siempre debemos poner el número de línea "a mano".
A diferencia de los restantes equipos Drean Commodore, la 64 no dispone del comando AUTO el cual genera automáticamente los números de línea, evitando tener que tipearlos a cada rato.
Este listado es, justamente, el comando AUTO.
Carguen el programa correspondiente. Antes de darle RUN asegúrense de grabarlo ya que éste termina con un violentísimo NEW.
Luego sí, podrán hacer RUN. A partir de aquí el programa estará listo. Para activar el nuevo comando, sólo deben tipear

DESCRIPCION DEL PROGRAMA

Línea	Descripción
100	Inicializamos la variable BASE con el valor de la dirección $C000. A partir de aquí se cargará el programa.
110	Determinamos, en función de BASE, el valor en donde se encontrara el módulo principal del programa.
120-160	Carga de las instrucciones en lenguaje máquina.
170	Actualizamos la dirección del módulo principal.
180	Modificamos puntero para atender al nuevo comando.
1000-1090	Sentencias DATA con las instrucciones en decimal.

VARIABLES UTILIZADAS:

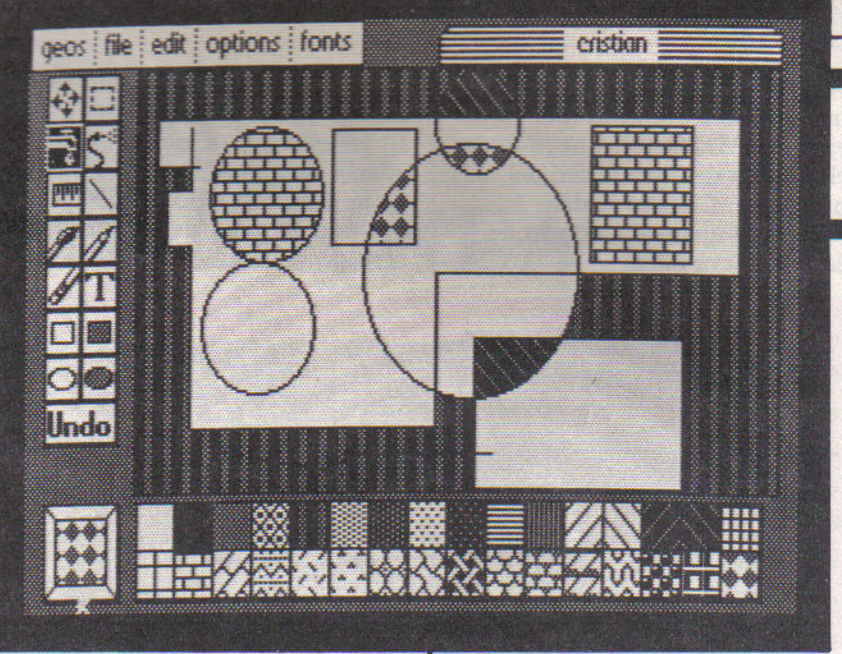
BASE:	Indica comienzo de carga.
WE:	Representa inicio del módulo principal del programa.
K:	Variable índice
N:	Toma las instrucciones en decimal.
C:	Sumatoria de todas las instrucciones.
X2, X1:	Parte alta y baja de la dirección de comienzo del módulo principal.

AUTO I donde I representa el incremento entre línea (por ejemplo AUTO 10). Luego tipeen el número de línea inicial del programa. Cuando le den RETURN aparecerá debajo de ésta la nueva línea.

Para desactivarlo sólo deberán tipear AUTO sin parámetros. El programa está escrito en código máquina y ocupa a partir de la dirección 49152 ($C000). En caso de que ustedes tengan otro programa almacenado a partir de esta dirección, cambien el valor de la variable BASE con la nueva dirección de carga. Automáticamente el programa cambiará todas sus referencias teniendo en cuenta la nueva dirección.

```
100 BASE=49152:REM A PARTIR DE $C000
110 WE=BASE+11
120 PRINT:PRINT" CARGANDO VALORES. UN MOMENTO..."
130 FORK=BASETOBASE+186:READN:POKEK,N:C=C+N:NEXT
140 IFC<>17067THENPRINT"ERROR EN DATAS. VERIFIQUE LOS
VALORES":STOP
150 PRINT"AUTO ACTIVADO. PARA ACTIVARLO TIPEE AUTO SEGUIDO DEL
INCREMENTO"
160 PRINT"PARA DESACTIVARLO SOLO TIPEE AUTO"
170 X2=INT(WE/256):X1=WE-X2*256:POKEBASE+1,X1:POKEBASE+6,X2
180 SYSBASE:NEW
1000 DATA 169, 11,141,  2,  3,169,192,141,  3,  3, 96
1005 DATA 173, 54,  3,208,  5,173, 55,  3,240, 49,173
1010 DATA  52,  3,240, 44,173, 56,  3, 24,109, 54,  3,141
1015 DATA  54,  3,173, 57,  3,109, 55,  3,141, 55,  3
1020 DATA 174, 54,  3, 32,205,189,169, 32, 32,210
1025 DATA 255,169,  2,133,198,169,145,141,119
1030 DATA   2,169, 17,141,120,  2, 32, 96,165,134
1035 DATA 122,142, 58,  3,132,123,140, 59,  3, 32
1040 DATA 115,  0,144, 67,240, 65,201, 65,208, 84
1045 DATA  32,115,  0,201, 85,208, 77, 32,115,  0
1050 DATA 201, 84,208, 70, 32,115,  0,201, 79,208
1055 DATA  63,169,  0,141, 52,  3,141, 54,  3,141, 55
1060 DATA   3, 32,115,  0, 32,107,169,165, 20,141
1065 DATA  56,  3,240,  3,141, 52,  3,165, 21,141, 57,
1070 DATA 240,  3,141, 52,  3, 76,116,164, 32,107
1075 DATA 169,208,  8,141, 54,  3,141, 55,  3,240, 10
1080 DATA 165, 20,141, 54,  3,165, 21,141, 55,  3
1085 DATA 174, 58,  3,172, 59,  3, 76,134,164
1090 DATA  50, 72,131,154,157,186
```

GEOS



Rating total: A
Creatividad: A
Documentación: B
Valor en relación al precio: Se justifica
Mantiene el interés: Sí
Computadora: Drean Commodore 64
Editor: Berkley Software

GEOS es un nuevo sistema operativo desarrollado para la Commodore 64. Como mencionamos en nuestra sección NEWS, también es compatible con la Drean Commodore 64.
Pero, ¿qué es el GEOS?
Son las siglas de Graphic Environmet Operating System.
Aquí cada comando o instrucción se reemplaza por un dibujito (llamado Icon). Basta con posicionarse con el mouse o el joystick para seleccionar una determinada función.
El término icon viene del inglés y significa ícono. Es una imágen pintada y dorada que se utilizó en Rusia y en la iglesia griega para representar a las Divinidades, santos, etc.
Antes de comenzar la descripción de este nuevo producto, queremos decirles que hacerla completa llevaría más de una página, que es lo que asignamos normalmente a los programas de "Revisión de Software".
Por este motivo, y debido fundamentalmente a la calidad del GEOS, es que en el próximo número realizaremos un estudio más profundo del mismo.
Ya sabemos lo que quiere decir GEOS. Lo que nos falta saber es para que sirve.
Por ahora relataremos una sesión de trabajo usando una Drean Commodore 64, la 1541 y el GEOS.

A través de un Joystick o del mouse, podemos realizar distintas tareas. Por ejemplo, redactar texto por medio del GEOWRITE o desarrollar gráficos usando el GEOPAINT.
El primero de ellos es un procesador de texto sumamente potente. El segundo es uno de los tantos utilitarios del GEOS, el cual nos permite realizar los más diversos gráficos.
También se nos suministran distintos accesorios. Por ejemplo, una calculadora, un reloj alarma y un anotador.

Cada uno de ellos representado con su icon respectivo. Así, el reloj se representa a través del típico despertador, la calculadora a través de una maquinita de bolsillo y el anotador como, justamente, un anotador.

De esta manera, y en cualquier momento, podemos utilizarlos. Por ejemplo, antes de comenzar a trabajar con el GEOWRITE, ponemos el reloj en hora.

Para ello solo basta posicionarnos sobre el icon del reloj y oprimir rápida y consecutivamente el botón del joystick dos veces.
Sobre la pantalla se abre una ventanita en donde aparece el reloj alarma.

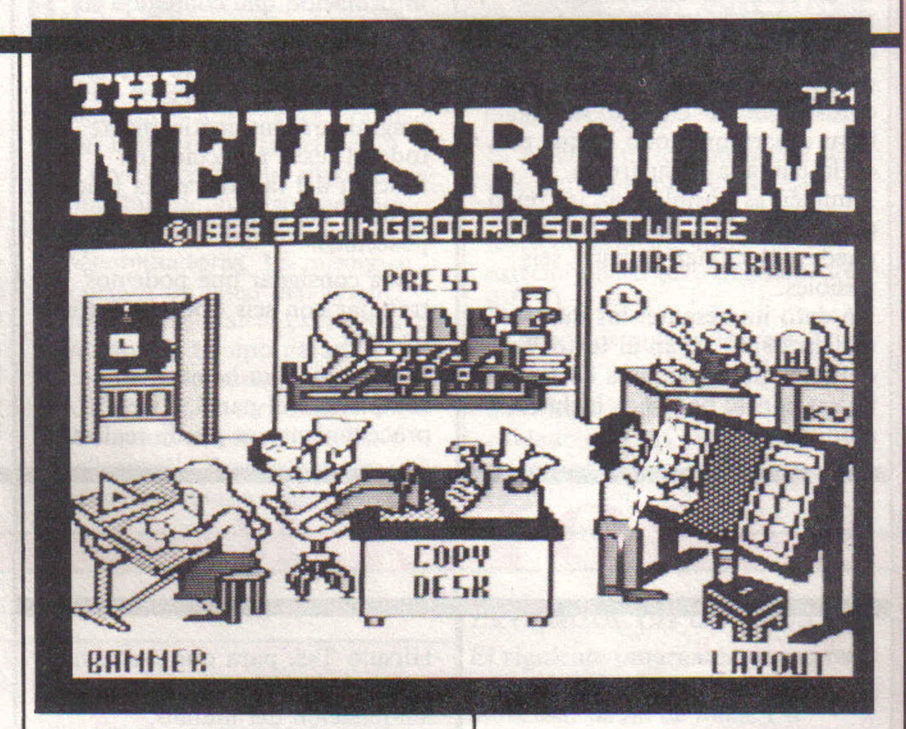
Luego de ponerlo en hora, oprimimos el "botón" en el reloj el cual saca la ventanita volviendo, así, a nuestra antigua tarea.
Otro de los accesorios es la calculadora ¿Para qué sirve?.
En casi todos los procesadores de texto para la Drean Commodore 64 sucede que a la hora de hacer algún tipo de cálculo tenemos necesariamente que tomar la calculadora y hacer la cuenta necesaria, ya que no podemos usar el PRINT del Basic.
Imagínense lo siguiente: estamos escribiendo un texto a través del GEOWRITE y en un determinado momento deseamos hacer una cuenta de sumas, divisiones, restas, raíz cuadrada, etc.

En la parte superior de la pantalla hay determinadas funciones, cada una con sus respectivos items.
Para este caso nosotros nos posicionaremos sobre la función GEOS, la cual suministra los distintos accesorios.
A través de Joystick tomaremos esta función oprimiendo el botón de disparo. En la pantalla aparece otra ventanita con cada uno de los accesorios.
Nos colocamos sobre la calculadora y oprimimos el botón. Ahora aparecerá en el medio de la pantalla otra ventanita pero con la calculadora de bolsillo.
Desde ya todos los gráficos están hechos en alta resolución. Otra

"bestialidad" es el GEOPAINT con el cual podemos realizar cualquier tipo de gráficos.
Con el GEOS es como si trabajáramos en un escritorio en donde hay una máquina de escribir, un anotador, un reloj alarma y un tablero con reglas y compases para dibujar.
Con este soft todo esto lo tenemos en la computadora.
Pero, lamentablemente, nos tendrán que disculpar por esta escasa descripción. Sin embargo la promesa ya fue hecha. En el próximo número, una nota dedicada exclusivamente al nuevo sistema operativo GEOS.

THE NEWSROOM

Rating Total: B
Creatividad: A
Documentación: B
Valor de relación al precio: Se justifica
Mantiene el interés: Sí
Computadora: Drean Commodore 64
Editor: Spr'agboard



NEWSROOM quiere decir "sala de periodistas". Justamente la finalidad de este poderosísimo utilitario es la de confeccionar todo tipo de boletines, hojas informativas, etc.
Desde ya para poder utilizarlo se necesita una impresora compatible.
La presentación de NEWROOM es sumamente divertida. Cada una de las funciones que se pueden realizar están descriptas a través de gráficos.
Estas son laboratorio de foto, titulera, redacción, diagramación, servicio de teletipo y finalmente la imprenta.
Cada una de ellos está representado respectivamente por un cuarto oscuro, una señorita escribiendo títulos, un redactor con las piernas sobre el escritorio leyendo el diario, un hombre diagramando las páginas, el operador de telex y la imprenta.
A través de cada una de ellas se pueden realizar los más diversos folletos, boletines, etc.
Por ejemplo podemos, primero, elegir la foto que corresponderá a un panel (cada página está formada por varios paneles o secciones). Estas se pueden elegir de un grupo ya realizadas y que acompañan al utilitario en discos

separados (calculamos que hay aproximadamente 700 gráficos distintos) o hacerlas nosotros mismos.
Para ello el software suministra herramientas de trabajo para dibujar, realizar líneas, círculos y cuadrados.
También dispone de 10 tramas distintas para pintar cualquiera de las zonas seleccionadas.
El trazo del lápiz se elige de entre 10 distintos. Si no estamos conformes con el dibujo, basta con seleccionar el tacho de basura para que todo lo que esté en la pantalla desaparezca.
También es posible mezclar texto con dibujos. La tipografía se puede seleccionar de entre seis posibles.
Un dato interesante: los dibujos realizados no tapan al texto. Si optamos por elegir una de la 700 ilustraciones, podemos inclusive modificarla.

Luego de completar el dibujo que constituirá nuestra imagen, tomamos la cámara y le sacamos una foto (se puede sacar la parte que querramos, como si fuese una cámara de verdad).

Finalmente grabamos nuestra foto y tomamos el módulo redactor. De esta manera comenzamos a redactar la información que contendrá un panel.

Luego ponemos sobre éste la foto. El NEWSROOM diagramará automáticamente todo el texto redactado en función del tamaño de la placa, es decir del espacio que poseemos.

Cabe consignar que podemos trabajar con seis tipos distintos de letra.

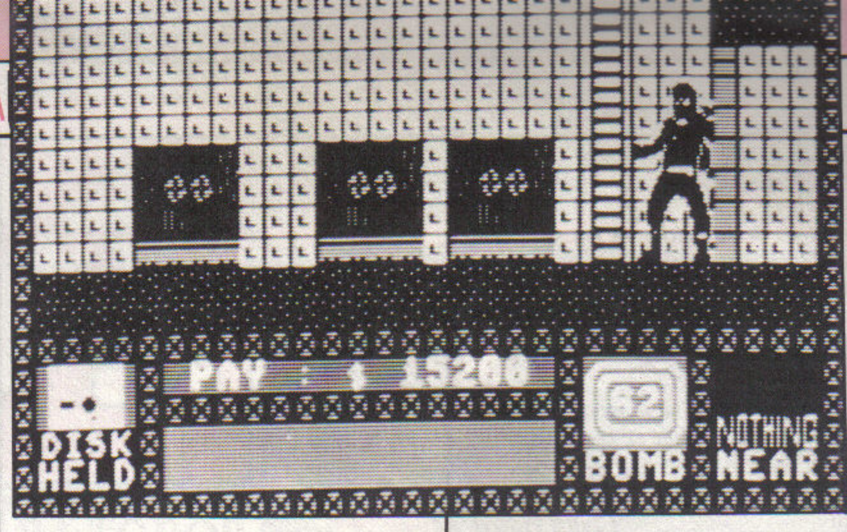
De esta manera hemos completado el panel. Este procedimiento se puede realizar para cada uno de los paneles.
Nos falta tomar al diagramador y decirle en qué lugar de la página queremos que aparezcan los paneles creados.
Sólo nos falta acceder a la imprenta e imprimir la página deseada.
Si queremos que nuestra página tenga un título, basta con llamar a la "titulera" (banner). Este módulo nos permitirá escribir él o los títulos deseados.
Cada una de las funciones son seleccionadas a través de la tecla Commodore junto con las que mueven el cursor. También es posible utilizar el joystick.
Todos los gráficos son realizados en alta resolución. Gracias a las grandes posibilidades de este utilitario es posible darles aplicaciones serias.
Por ejemplo, algunos lo utilizan para preparar publicidad, boletines informativos, etc.

SABOTEUR

Rating Total: B
Creatividad: B
Documentación: B
Profundidad del juego: B
Valor en relación al precio: Se justifica
Mantiene el interés: Si
Computadora: Drean Commodore 64
Editor: Durell

Transcurre el año 1999. En la isla de Tokiparatonki se establece el cruel y despiadado emperador Danito, quien quiere concretar un viejo anhelo: destruir al mundo.
Para ello cuenta con un sofisticadísimo cuartel, donde se encuentra la computadora que hará lanzar las bombas atómicas hacia las naciones del mundo.
Estas, angustiadas por el peligro, deciden enviar a la isla del emperador al agente mundial Hitsuto Tse, para que destruya la computadora evitando, así, la aniquilación del mundo.
Pero la tarea no será fácil. No dispone de armas de fuego. Por el contrario, sus armas sólo serán aquellas que pueda encontrar en el cuartel.
Aprovechando una negrísima noche, nuestro hombre consigue acceder al embarcadero del cuartel, utilizando un bote de goma. El primer arma que tiene es una estrellita envenenada, la que será lanzada si nosotros oprimimos el botón del joystick.
A partir de aquí el tiempo comienza a disminuir teniendo, aproximadamente, cinco minutos para poner la bomba en la sala de computadoras.
El acceso a esta sala no es para nada fácil. Se encuentra en la segunda planta subterránea del cuartel.
Nosotros debemos guiar a Tse a través de las distintas galerías subterráneas buscando los transportes que nos trasladarán de una planta a otra.
Nuestro agente no tiene la bomba en su mochila. Debe encontrarla en uno de los almacenes de la segunda planta.
Las tropas del emperador Danito son los mortales perros guardianes y los especialistas en Kung-Fu.
Todos tratarán a toda costa de evitar nuestro arribo a la sala de computadoras. Los primeros con sus dolorosas mordeduras y los otros a través de armas envenenadas acompañadas de feroces patadas.
Tse dispone de un indicador que le dice al jugador cuanto de vida le queda. Cada mordedura o patada la disminuye.
Si el indicador llegara a cero, la vida de Tse se acabará instantáneamente, pudiendo el emperador Danito destruir el mundo.

También el jugador que guiará a nuestro agente sabe que es lo que ve Tse dentro del cuartel. De esta manera le será posible recoger del suelo armas que luego utilizará, como cuchillos, piedras, etc.
Si somos capaces de instalar la



bomba, para lo que habremos tenido que llegar a la segunda planta subterránea, tendremos que huir rápidamente hacia la terraza en busca del helicóptero salvador. Tampoco disponemos de mucho tiempo. El mecanismo detonador de la bomba tiene un reloj interno.
Dentro del cuartel también hay unos dispositivos electrónicos que lanzan balas. Cada una de estas balas provoca que la vida de Tse disminuya.
Por suerte podemos recuperarnos. Basta con quedarnos unos momentos quietos para que el indicador de vida suba hasta el tope.
No debemos descuidar el tiempo que disponemos para cumplir con la misión, el cual disminuye estando o no quietos.
La bomba, como dijimos antes, se encuentra en uno de los almacenes de la segunda planta. Cuando pasemos sobre ella aparecerá sobre el indicador que denota lo que "ve" Tse, una cajita con el nombre de Bomb. Aquí debemos pulsar el botón del joystick para que ingrese en la mochila de nuestro agente y dirigirnos rápidamente a la sala de computadoras. La misma se encuentra un poco más abajo. Luego de instalar el explosivo, el contador de tiempo pasará a rojo y comienza la cuenta regresiva.
SABOTEUR es un juego sumamente entretenido como "peligroso". Tendrá más de cincuenta pantallas, las que representan todos los posibles lugares en que puede estar Tse.

El nivel de dificultad puede ser seleccionado entre los nueve posibles; desde extremly easy (extremadamente fácil) hasta extremly hard (extremadamente difícil).

A nosotros nos bastó con el primero.

Solo les advertimos lo siguiente: cuidado con los perros. ¡No están vacunados contra la rabia!!!

TRUCOS

```
10  FORJ=1TO54:READK:CS=CS+K:NEXT
20  IFCS<>5908THENPRINT"ERROR EN DATAS. VERIFIQUE LOS
    VALORES":STOP
30  PRINT"DATOS OK. BORRE LAS LINEAS 10,20,30 Y 40. LUEGO
    ALMACENE EL PROGRAMA"
40  PRINT"EN CASSETTE O DISCO. FINALMENTE TIPEE RUN 100 Y
    NEW.":END
100 REM AVISADOR FIN DE LINEA
110 FORJ=0TO53:READK:POKE49152+J,K:NEXT
120 SYS49193
130 DATA 169,009,141,005,212,141,006,212
140 DATA 169,110,141,001,212,169,015,141
150 DATA 024,212,169,017,141,004,212,169
160 DATA 016,141,004,212,096,165,211,201
170 DATA 075,208,003,032,000,192,076,049
180 DATA 234,120,169,029,141,020,003,169
190 DATA 192,141,021,003,088,096
```

Avisador fin de línea

El siguiente programa, escrito en lenguaje máquina, es un indicador de fin de línea. En realidad el programa emite un leve beep cuando estamos en la columna 76. Lo podríamos comparar con la campanita de la máquina es escribir.
De esta manera podemos evitar seguir tipeando líneas de programa cuando lleguemos a esta columnas.

GET A$

Generalmente cuando debemos esperar que se orpima alguna tecla para seguir ejecutando un programa, utilizamos la sentencia GET de la siguiente manera:
100 PRINT"OPRIMA UNA TECLA PARA SEGUIR"
110 GETA$:IFA$=""THEN 110
120 CONTINUA EL PROGRAMA
Otra forma de realizar ésto es:
100 PRINT"OPRIMA UNA TECLA PARA SEGUIR"
110 WAIT198,1:GETA$
De esta manera nos evitamos el IF.

DATASSETTE

Ante todo deseo felicitarlos por su excelente revista. Me gustaría que me respondan las siguientes inquietudes:
1) ¿Qué procedimiento debo hacer para que aparezca el juego del cassette en la pantalla?
2) Cuando culmino un programa, ¿cómo hago para que se grabe en el cassette?

Enrique Boschetti
Capital Federal

Para cargar un juego en tu computadora debemos realizar el siguiente procedimiento:
LOAD "nombre del programa"
Por ejemplo, si quieres cargar desde el cassette el Girus, tienes que hacer LOAD "GIRUS" y luego oprimir la tecla RETURN.
Para grabar tu programa en cassette debes hacer:
SAVE "nombre del programa"
Por ejemplo, si quieres grabar el programa "ABCD" debes hacer SAVE "ABCD" y luego oprimir la tecla RETURN.

INTERCAMBIO

Para empezar deseo felicitarlos por la magnífica revista que crearon y de la que soy un asiduo lector tanto para entretenerme como para consultarla.
Estoy interesado en realizar intercambio de programas y/o datos con otros usuarios de la Drean Commodore 64. Mi nombre es Ivan Mantovani y mi dirección está más abajo.
También me dirijo a ustedes para hacerles llegar unas preguntas y es que poseo una Commodore americana y desearía saber si sirve el drive de la Drean Commodore 64 y también si sirven los diskettes.
Otra pregunta es como puedo hacer para averiguar un SYS, pues el del juego llamado "Falcon Patrol" lo he olvidado.
Sin más, me despido y los saludo atentamente agradeciéndoles su atención y la oportunidad que le brindan al lector de poder comunicarse con nosotros.

Iván Mantovani
Cerros Colorados 1432 - (1828) Banfield

Todos los periféricos (inclusive los diskettes) que se encuentran disponibles para la Commodore 64 son cien por cien compatibles con la Drean Commodore 64.

Continuamos con esta sección para que los lectores planteen sus consultas y sugerencias. Para eso deben escribir a Revista para usuarios de Drean Commodore, Paraná 720, 5to. Piso, (1017) Cap.

Lo mismo ocurre al revés, es decir que todos los periféricos para la Drean Commodore 64 son compatibles con la Commodore 64. Con esto contestamos tu primera pregunta.
Para determinar la dirección inicial de un programa almacenado en disco (comúnmente llamado averigua SYS) te dejamos el siguiente programa que imprimirá la dirección en memoria en donde el programa comienza a almacenarse.

```
10 INPUT "NOMBRE DEL PROGRAMA" ; NP$
20 OPEN2,8,2,NP$+",P,R"
30 GET#2,LO$:LO$ = LO$+CHR$(0)
40 GET#2,HI$:HI$ = HI$+CHR$(0)
50 CLOSE2:DI=ASC(LO$) + 256*ASC(HI$)
60 PRINT "LA DIRECCION INICIAL ES:";DI
70 END
```

COMMODORE IMPORTADA

Poseo una Commodore 64 modelo americano, adquirida en Estados Unidos hace tres meses, pero por anteriores experiencias con otras máquinas (léase Apple, Sinclair, etc) y un Basic bastante amplio, domino bastante la computadora y me gustaría trabajar con ella.
Su revista me parece excelente, ya que trae notas avanzadas (todas las que se refieren a código máquina), notas para principiantes, etc. Los felicito, no bajen el nivel. Además tengo las siguientes dudas:
1) He comprobado que varios juegos (entre ellos Comando, Misión Imposible, Kung fu Master) no corren en el modelo nacional, no teniendo problemas en el modelo importado. ¿A qué se debe?
2) Algunos modelos de la C-64 ejecutan el sonido (sobre todo las melodías) mucho más lentamente que otros, ¿por qué sucede esto? (personalmente supongo que se debe a variaciones del chip SID).
3) Necesitaría me dieran la dirección que controla la velocidad del parpadeo del cursor, así como un POKE que mantenga el cursor fijo (sin destellar). Se que en la C-128 esto se consigue mediante ESC E.
4) En una revista europea he leído que la C-128 en modo 64 tiene dos posiciones de memoria que difieren de las del C-64 original (concretamente en las posiciones 0 y 1 de memoria, según dice la publicación; si hacemos un PRINT PEEK (1) en la C-128 modo 64 responderá 119 mientras que el C-64 responderá 55). ¿Es cierto esto?

Juan Gutmann
Capital Federal

Con respecto a la primera pregunta, es la primera vez que escuchamos que programas para la Commodore 64 no funcionen en la Drean Commodore 64.
Tal vez lo que nos cuentas se produjo debido a que esos programas han sido probados en una Drean Commodore defectuosa, lo que no implica que las restantes estén así.
Asimismo, si ejecutamos una melodía en cien Drean Commodore 64 que funcionen correctamente, se trendrían que oir de la misma manera en cada una (también pudo suceder que operases sobre máquinas defectuosas).
Las direcciones pedidas son: 205 gobierna el parpadeo del cursor mientras que la 204 sí está en modo flash o en modo continuo.
Finalmente, te diremos que es cierto que el contenido de la dirección 1 varía en la 128 como en la 64. Esto se debe a que esa dirección le indica al sistema operativo qué áreas son RAM y cuáles son ROM. Como la C-128 dispone de mayor memoria libre, este valor es diferente al de la 64, la cual tiene menor RAM.

AGOSTO 1986 Nº 17 ₳ 2,50 REP. ARGENTINA

K64

COMPUTACION PARA TODOS

Tecnología Informática Nacional

Software Médico

Cómo se Almacenan los Datos

Experiencias de los Usuarios

Computación Para Todos Los Docentes: Suplemento Para CZ, TK, TI, TS, C-64 y MSX

EL HOMBRE, LA EDUCACION Y LA INFORMATICA.

"La conjunción entre educación y progreso técnico se proyecta al servicio del hombre"

1er. CONGRESO ARGENTINO DE INFORMATICA EDUCATIVA



BUENOS AIRES, SHERATON HOTEL 22 AL 26 DE SEPTIEMBRE 1986

Organizado por el

INSTITUTO ARGENTINO DE INFORMATICA

Asociación Civil sin fines de lucro
Inscripto en el Registro Nacional de Entidades de Bien Público N° 2864
FUNDADO EN 1966 — PERSONERIA JURIDICA: Resolución I.G.J. 6335/71
Avda. ROQUE SAENZ PEÑA 651 — (1035) BUENOS AIRES — Teléfonos: 46-2827/6585

Está dirigido a todos los que ejercen la instrucción en sus diversos niveles -desde el elemental hasta el posgrado-, a autoridades educacionales oficiales y privadas, pedagogos, educadores, especialistas, además de profesionales, técnicos, alumnos, padres y público en general.

Los asistentes dispondrán de los elementos más actualizados en innovaciones tecnológicas, experiencias e instrumental informático y recibirán los Anales del Congreso con material inédito seleccionado por el Consejo Académico.
En sesiones de taller, diariamente, se desarrollarán trabajos prácticos que ilustrarán cada tema.

Auspiciado por

UNESCO • Ministerio de Educación y Justicia de la Nación • Secretaría de Desarrollo Humano y Familia de la Nación • Secretaría de Comunicaciones de la Nación • Dirección General de Escuelas y Cultura de la Provincia de Buenos Aires • Otros Organismos Provinciales de Educación y Cultura • CONSUDEC • Universidad Católica de Tucumán • Universidad Católica de Salta • Otras Universidades e Instituciones Privadas.

Autoridades

CONSEJO ACADEMICO: Prof. Alfredo **VAN GELDEREN**; Dr. Luis **SANTALÓ**; Dr. Oscar **GÓMEZ POVIÑA**; Dr. Alberto **TAQUINI (h)**; Rev. Hno. Septimio **WALSH**; Dr. Avelino J. **PORTO**; Dr. Horacio E. **BOSCH**; Prof. Icas J. **MICILLO**; R.P. Michel **RAMLOT O.P.**; R.P. Cesáreo **CAMPOS S.D.B.**; Prof. Juan Martín **BIEDMA**; Dr. Alberto **OBLIGADO NAZAR**; Prof. Héctor O. **PUEYO**; Dr. Aldo Armando **COCCA**; Prof. Nelly **V. de TAPIA**; Ing. Edmundo N. **GRAMAJO**; Dr. Juan Carlos **ANSELMI**; Prof. Sergio **PAPA BLANCO**; Dr. Horacio H. **GODOY**; Ing. Eitel H. **LAURÍA**; Dr. Adolfo **PORTELA**; Prof. Arturo de la **ORDEN HOZ**; Dr. Antonio **ARAMOUNI**.

CONSEJO EJECUTIVO: Dr. Enrique **DIEULEFAIT**; Dr. Rodolfo **STALANICH**; Ing. Ana María **VELO**; Prof. Carlos **TAPIA**; Dr. Gabriel **ARAMOUNI**; Lic. Alfredo **GUIROY**; Lic. Jorge A. **REY VALZACCHI**; Prof. María C.W. de **GARCÍA JUANICÓ**; Lic. Carlos A. **GRILLÉ**; Dr. Roberto **BONANNI REY**; R.P. Jorge **YIGUERIMIAN**; Prof. Rodolfo **DALVARADE**; Prof. Ana María **POLLITZER**; Dr. Jorge O. **ALIJO** Dr. Alfredo **PÉREZ ALFARO**; Dr. Horacio R. **GRANERO**; Dr. Gerardo C. **LUPPI**; Dr. Nevio J. **BORRONE**; Prof. Lorenzo **CÁFFARO**; Prof. Alicia **GAGO**; Lic. Angel **de ESTRADA**; Dr. Waldemar J. **SAÚC**; Lic. Fernando H. **PETRONI**.

GUIA TEMATICA DEL CONGRESO (Síntesis)

Las áreas básicas de análisis serán las siguientes:

(I) Formación del Pensamiento Lógico.

(II) Demandas científicas y tecnológicas de la sociedad contemporánea.

(III) Estado actual de la Informática Educativa para los distintos niveles, ciclos y modalidades.

(IV) Prospectiva dentro y fuera del sistema educativo.

(V) Recursos Humanos y Materiales.

Los trabajos que se presenten deberán responder a la siguiente guía temática:

(1) Requisitos humanísticos de la Informática Educativa.

Reafirmación de los valores trascendentes del ser humano en la inserción tecnológica. Rol de la computación en el desarrollo de nuevas pautas culturales. Recaudos.

(2) Desarrollo de la Inteligencia.

(3) Nuevas metodologías alfabetizadoras. Lenguajes de programación para ia creatividad.

(4) Didácticas especiales para cada nivel de educando.

(5) Educación para alumnos discapacitados.

(6) Desarrollo curricular y educación informática para docentes de los distintos niveles. Desarrollo y aplicación de software interactivo para la currícula estudiantil.

(7) Educación grupal a padres sobre potencial educativo y utilización escolar de las microcomputadoras.

(8) Multimedios Informáticos.

(9) Sistemas Educativos No Convencionales. Teleinformática Educativa. Integración federal.

(10) Banco de Datos Educativo.

(11) Evaluación pedagógica de proyectos educativos.

(12) Capacitación y Desarrollo Informático para optimizar el ejercicio y gestión de las profesiones: Informática Administrativa/Informática Médica/Informática Jurídica y Notarial/ Informática Agropecuaria/Informática de la Ingeniería.

(13) Inteligencia Artificial.

(14) Protección legal del software educativo.

(15) Administración educacional computadorizada. Control de Costos de la Informática Educativa.

(16) Vulnerabilidades de la Informática Educativa. Delitos. Auditoría y Seguridad del Sistema.

(17) Política Nacional en Informática Educativa. Necesidades de la educación argentina. Actualidad y prospectiva.

Régimen de Becas para Instituciones Oficiales y Privadas.

Abierta la inscripción — Recepción de trabajos: hasta el 31/8/86, previa Solicitud del Reglamento

Los interesados en inscribirse y en recibir orientación estrictamente académica sobre bibliografía, material didáctico especializado y programas de capacitación docente para distintos niveles, deben dirigirse personalmente o por correo a: Sr. Presidente del INSTITUTO ARGENTINO DE INFORMATICA, **Dr. Antonio Aramouni**, Av. Roque Sáenz Peña 651 - (1035) Buenos Aires.